AutoMax
Ladder Language Editor

Instruction Manual J2-3093-3

RELIANCE ELECTRIC

The information in this user's manual is subject to change without notice.

DANGER

ONLY QUALIFIED ELECTRICAL PERSONNEL FAMILIAR WITH THE CONSTRUCTION AND OPERATION OF THIS EQUIPMENT AND THE HAZARDS INVOLVED SHOULD INSTALL, ADJUST, OPERATE, OR SERVICE THIS EQUIPMENT. READ AND UNDERSTAND THIS MANUAL AND OTHER APPLICABLE MANUALS IN THEIR ENTIRETY BEFORE PROCEEDING. FAILURE TO OBSERVE THIS PRECAUTION COULD RESULT IN SEVERE BODILY INJURY OR LOSS OF LIFE.

WARNING

THE USER MUST PROVIDE AN EXTERNAL, HARDWIRED EMERGENCY STOP CIRCUIT OUTSIDE THE CONTROLLER CIRCUITRY. THIS CIRCUIT MUST DISABLE THE SYSTEM IN CASE OF IMPROPER OPERATION. UNCONTROLLED MACHINE OPERATION MAY RESULT IF THIS PROCEDURE IS NOT FOLLOWED. FAILURE TO OBSERVE THIS PRECAUTION COULD RESULT IN BODILY INJURY.

Preface

AutoMax Ladder Editor Feature Overview

The AutoMax Ladder Editor ships as part of the AutoMax Programming Executive. It can be run from within the Executive or as a stand-alone application for offline programming only. The Editor provides these features to help you create your ladder programs:

More easily view and work with rungs in the same program or in different programs.

Because the Editor is based on Microsoft Windows 95™, you can:

- Use familiar menu commands and toolbar icons for typical Windows features like restore, maximize, open, save, cut, copy, and more
- Display multiple rungs at the same time
- Display rungs from different parts of the program at the same time
- Display rungs from more than one program at the same time
- Use pop-up menus to access frequently used commands

More easily edit ladder programs.

Because of the drag-and-drop functionality and property sheets for each program element you can:

- Cut, copy, paste individual instructions and rungs or groups of instructions and rungs
- Move rungs by selecting them and dragging them with the mouse
- Create a “work-in-progress program” by building rungs without variable names or coils when you end an editing session
- Insert instructions and branches, or change an instruction type or variable name without any re-typing
- Reduce the amount of typing by importing the variable descriptions for global variables from the Variable Configurator. Also, while entering variable names, you can choose from a list of smart-matched names
- Insert new rungs without renumbering existing rungs
- Add rung descriptions which become part of the rung

Create programs using an enhanced set of ladder instructions.

The AutoMax Ladder language has been enhanced to include support for:

- An expanded instruction set based on the IEC-1131-3 standard
- Variable names of up to 16 characters
- Multiple coils per rung
- More instructions per rung and more parallel branches
- Easier access to bits within integers and arrays, for example: INPUT_VAR.31 or array_var[index].bit_num

More easily edit online programs.

When editing an online program you can:

- Identify inserted, deleted, or modified rungs
- View the original program while you are editing an online version of the same program
- Monitor and modify timer and counter preset values
- Test online edits before permanently downloading them to the Processor
- Set, force, and unforce variables from within the Editor

Where to Find More Information

This instruction manual describes the AutoMax Enhanced Ladder Editor. For information about the Ladder language, refer to instruction manual J2-3094. The information in both instruction manuals can also be accessed in the AutoMax Ladder Editor and Enhanced Ladder Language help file (MAXED.HLP). The help file contains additional information not found in either instruction manual.

For more information about the AutoMax Programming Executive, refer to J2-3089.

The thick black bar shown at the right-hand margin of this page will be used throughout this instruction manual to signify new or revised text or figures.

Table of Contents

1.0 Getting Familiar with the Editor 1-1
1.1 Overview of the AutoMax Ladder Editor Window 1-1
1.2 Opening Programs 1-2
1.3 Closing Programs 1-3
1.4 Saving Programs 1-3
1.5 Saving Programs Automatically As You Work 1-4
1.6 Displaying a Program in Multiple Windows 1-4
1.7 Editing Different Parts of the Same Program Using Split Windows 1-5
1.8 Editing Multiple Programs 1-5
1.9 Displaying/Hiding Revision Marks While Editing Programs Offline 1-6
1.10 About Program Properties 1-7
1.11 About Program Execution 1-7
1.11.1 Specifying a Scan Time to Control Program Execution . . 1-8
1.11.2 Specifying a Hardware Event to Control Program Execution 1-8
1.11.3 Specifying a Software Event to Control Program Execution 1-9
1.12 Running and Stopping Ladder Programs 1-10
1.13 Exiting the Ladder Editor 1-10

2.0 Editing Programs 2-1
2.1 Starting a Rung 2-1
2.1.1 Defining the Horizontal and Vertical Grid Limits Per Rung 2-2
2.1.2 Turning the Grid On and Off 2-2
2.2 Selecting Rungs 2-2
2.3 Entering Rung Descriptions 2-4
2.4 Inserting Instructions into a Rung 2-5
2.5 Connecting Instructions (Drawing Wires) 2-6
2.6 Connecting Multiple Instructions by Using the ENO Output Bit 2-6
2.7 Selecting Instructions 2-7
2.8 About Instruction Properties 2-8
2.9 Changing an Instruction Type 2-9
2.10 Assigning Variables and Constants to Ladder Instruction Parameters 2-10
2.11 Entering Variable Descriptions 2-11
2.12 Using Variable Smart-Matching 2-12
2.13 Selecting Variable Names 2-13
2.14 Changing a Variable's Data Type 2-13
2.15 Changing a Variable's Scope 2-14

3.0 Printing Programs 3-1
3.1 What Is Included in a Printed Copy of a Program 3-2
3.2 Setting Print Options 3-4
3.3 Defining the Page Setup 3-5
3.4 How a Rung Is Printed To Fit on a Single Page 3-6

3.5 Inserting and Deleting Page Breaks 3-7
3.6 Printing a Range of Rungs 3-7
3.7 Displaying and Printing Coils as Right-Justified 3-8
3.8 Printing Multiple Copies of Programs 3-8

4.0 Verifying Programs 4-1
4.1 Creating a Verify Error Log File 4-2
4.2 Automatically Checking the Variable Configurator Database While Verifying a Program 4-2
4.3 Specifying Whether To Include Warning Messages When Verifying Programs 4-3
4.4 Resolving Verify Errors 4-3

5.0 Editing Programs Online 5-1
5.1 About the States of Online Programs 5-2
5.2 Monitoring an Online Ladder Program 5-4
5.3 Pausing the Editor 5-5
5.4 About Power Flow 5-5
5.5 Monitoring Data in an Online Program 5-6
5.6 Accepting Changes Made to Online Programs 5-6
5.7 Downloading Changes Made to an Online Program (Commit Online Changes) 5-7
5.8 Testing Programs (Test Mode) 5-9
5.8.1 Committing an Online Program in Test Mode (Commit Changes in Test Mode) 5-9
5.8.2 Removing an Online Program from Test Mode 5-10
5.8.3 Canceling All Changes Made to an Online Program 5-10
5.9 About How Changes Made to Online Programs Are Verified . . . 5-11
5.10 Viewing an Online Program as It Looked before It Was Edited . 5-12
5.11 Capturing the State of Logic in an Online Program 5-13
5.11.1 Capturing the State of Logic Based on a Coil Becoming True 5-14
5.11.2 Capturing the State of Logic Based on a Coil Becoming False 5-14
5.11.3 Capturing the State of Logic Based on a Rung Error 5-15
5.11.4 Waiting for Triggers While Continuing To Edit 5-15
5.11.5 Re-Activating a Trigger 5-15
5.11.6 Clearing a Trigger 5-16
5.12 Setting and Forcing Variables in Ladder Programs 5-16
5.12.1 Setting Variables 5-18
5.12.2 Forcing Variables 5-19
5.12.3 Unforcing Variables 5-20
5.12.4 About the Expanded Set/Force/Unforce Dialog Box 5-21
5.12.5 Viewing the Total Number of Forced Variables 5-21
5.12.6 Viewing a List of Set or Forced Variables 5-21
5.12.7 Unforcing an Entire Force Page 5-21
5.12.8 Testing If Variables in the Rack Are Forced 5-22
5.13 Viewing and Clearing Run-Time Errors 5-22

Appendices

Appendix A
Toolbars, Palettes, and the Status Bar A-1

Appendix B
Keyboard Shortcuts B-1

Appendix C
Using the Pre-Defined (Reserved) Ladder Language Variables C-1

Appendix D
Online Editing Memory Limits D-1

Appendix E
Rung Execution Order E-1

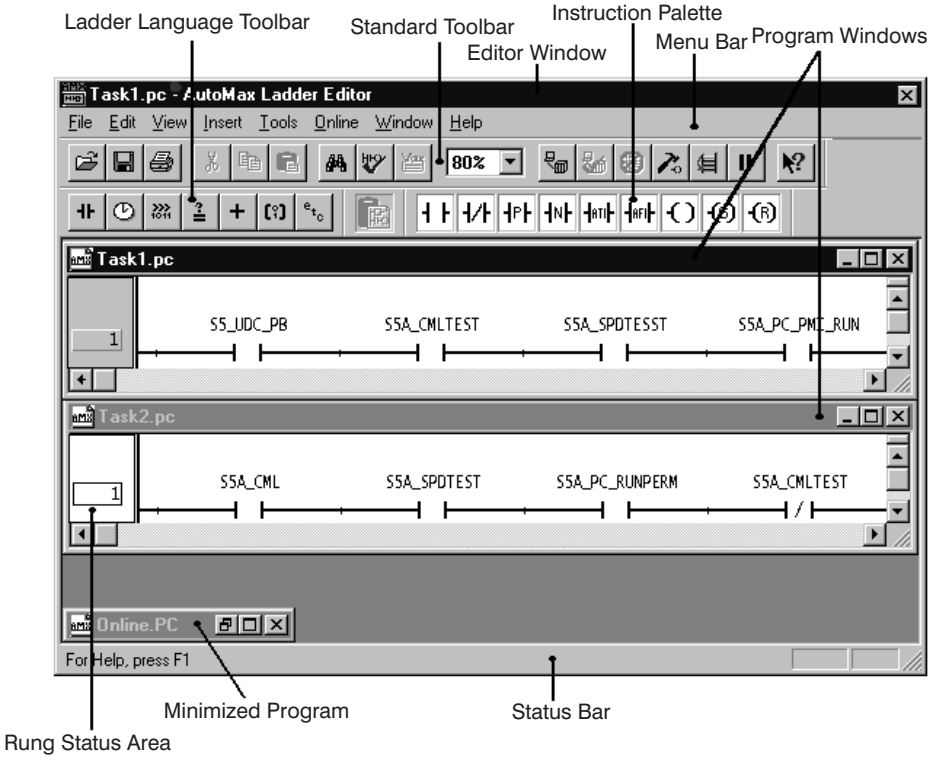
Appendix F
Converting Ladder Programs Created Using an Older Version of AutoMax F-1

Appendix G
Glossary G-1

1.0 Getting Familiar with the Editor

This section describes some basic information to help you become familiar with the Editor.

1.1 Overview of the AutoMax Ladder Editor Window



- **Ladder Language Toolbar:** Provides access to the individual ladder instruction palette
- **Standard Toolbar:** Provides shortcuts to common commands by clicking on a button
- **Editor Window:** Identifies the name of the currently active program and provides access to the application window controls
- **Instruction Palette:** Lets you drag and drop instructions into your program
- **Menu Bar:** Provides access to the available Editor commands, many of which can be accessed via a standard toolbar button or keyboard characters

- **Program Windows:** Presents the open program(s). The name of the program is listed in each window
- **Rung Status Area:** Displays rung numbers, revision marks, set triggers, and indicates active versus non-active powerflow
- **Minimized Program:** Can help organize many open programs
- **Status Bar**: Displays information about the program objects you have selected, the state of the program in the active window, and helpful messages

Refer to Appendix A for more information about the toolbars, instruction palettes, and the status bar.

1.2 Opening Programs

Opening a program is the first step in editing it. Before you can open a program, you must have added it to the rack using the Task Manager application.

You can open a program from the Task Manager application or from the AutoMax Ladder Editor.

To open a program from within the Task Manager application

- From the Task Manager application, select a task and choose Edit from the Task menu.

To open a program from within the Editor

Step 1. Do one of the following:

- Click on 

or

- From the Ladder Editor's File menu, choose Open

The Open dialog box is displayed.

Step 2. Choose the location of the file that you want to open. The default drive and directory is the drive and directory from which a program was last opened.

To change this:	Do this:
The drive and directory in which to search	Click in the Look in list box. From here choose the drive and directory that contains the program you want to open.
The directory in which to search	Click on a folder or on a file in the list of files and folders contained within a directory.

Step 3. From the list of file names, select the program you want to open. The default file type is ladder programs (*.pc). You can open only .pc programs using the Ladder Editor.

Step 4. Open the file by clicking OK.

If the ladder program was created using an older version ladder editor, the Editor first prompts you to convert the

program. To convert the program, choose Yes. For more information, see Appendix F.

You can have at most 15 unique offline programs opened at one time.

1.3 Closing Programs

Close the active program with the Close command. The Editor stays active and any other open programs stay open.

The Editor prompts you to save any changes to open programs before closing.

To close a ladder program

Step 1. From the File menu or Program Control Menu, choose Close.

Step 2. If you have made changes to the program since you last saved it, choose one of the following buttons in the message box that asks if you want to save the changes:

To:	Click:
keep the changes	Yes
not save the changes	No
keep the program open and cancel the changes	Cancel

1.4 Saving Programs

Save changes to the active, offline program by using the Save command. The Editor stays active with the same program open after saving the changes.

To save a ladder program

- Click on 

or

- From the File menu, choose Save

The Editor saves global and local variables and their descriptions to the program symbol table.

Tip

Any offline revision marks present in the program disappear once you save a program.

Tip

To save changes made to an online program, use the Accept or Commit Test Mode Changes commands. For more information, see Chapter 5.0.

1.5 Saving Programs Automatically As You Work

To make sure that you do not lose work, you can use the Automatic Save Every feature to save your offline ladder programs at a specified interval—for example, every ten minutes. Specify whether you want your programs to be saved automatically and the time interval to use within the General Options tab.

Files that are saved automatically are stored in the same directory as the original ladder program and under the same name. However, the file extension for an automatically saved file is .ASV.

To automatically save programs as you work

Step 1. From the Tools menu, choose Options.

Step 2. On the General tab, choose Automatic Save Every.

Step 3. Using the Minutes spinner box, choose or type the time interval (in minutes) that you want the Editor to wait before it automatically saves the ladder program. Choose an interval from 1 to 120 minutes. The default setting is 10 minutes.

The program is saved at your specified time interval whenever a change to the program has been made since the last automatic save.

If a program you are trying to open has a more current autosaved copy, the Editor prompts you to choose either the autosaved file or the program file you last saved.

The auto-saved copy of a ladder program is deleted whenever you save the program by using the Save command or close the program.

1.6 Displaying a Program in Multiple Windows

To help you simultaneously view different parts of a program, you can choose to display it in more than one window. Any changes you make to the program are reflected in the other window(s) in which the program is displayed. When you open a new window, it becomes the active window and is displayed on top of all other open windows.

To display a program in multiple windows

Step 1. Make sure that the program you want to display in a new window is the active program.

Step 2. Do one of the following:

- From the Window menu, choose New Window.

or

- With no program items selected and the mouse pointer positioned in the grid area, click the right mouse button, and choose New Window from the pop-up menu.

The program window name is appended by a ":#". For example, if you chose to display the program LINE_1.pc in a new window, the new program window would be LINE_1.pc:2.

1.7 Editing Different Parts of the Same Program Using Split Windows

A program window may be split into two horizontal panes. Any editing done in one pane is reflected in the other.

Each pane can be scrolled independently using its own vertical scroll bar.

To view different parts of the same program using the Split command

Step 1. From the Window menu, choose Split. The cursor turns into a split pointer, a combination up arrow and down arrow.

Step 2. Using the mouse or the up and down arrow keys, position the cursor at the point in the program where you would like to split the screen.

Step 3. Press ENTER or click the mouse button once to position the split between the panes.

To view different parts of the same program by clicking on the splitter bar

- While pointing to the splitter bar, press the left mouse button, drag the splitter bar to the desired location, and release the button.

To remove the splitter bar

- Click on the splitter bar and drag it to the top or bottom of the window.

or

- Double-click anywhere on the splitter bar.

1.8 Editing Multiple Programs

You can edit multiple programs (up to 15) within the Editor at the same time. Each program has its own window. Only one window at a time can be the active window, which you can identify by the highlighted title bar and rung status area.

To edit multiple ladder programs from the Task Manager

Step 1. From the Task Manager, select each ladder program that you want to edit.

Step 2. From the Task Menu, choose Edit. The files you selected are opened along with the Editor.

Step 3. From the Editor's Window menu, select Tile or Cascade. All non-minimized programs are arranged within the Ladder Editor window. Or, use the mouse to arrange the windows on the desktop.

Step 4. Click on the program window to make it active before editing the program.

To edit multiple ladder programs from the Editor

Step 1. From the File menu, open all the programs you want to edit.

Step 2. From the Window menu, select Tile or Cascade. All non-minimized programs are arranged within the Ladder Editor window. Or, use the mouse to arrange the windows on the desktop.

Step 3. Click on the program window to make it active before editing the program.

Tip

To switch among the programs you are editing, press CTRL+F6.

1.9 Displaying/Hiding Revision Marks While Editing Programs Offline

Revisions are always displayed while you are editing a program online. But you can choose to display offline editing revision marks.

Revisions are marked by a letter in the rung status area and by the modified rung color.

Letter	Meaning
M	Logic was modified, or wires have been added or deleted.
I	The rung has been inserted.
D	The rung has been deleted.

Tip

If you delete a newly inserted rung before committing or saving the change, the rung is deleted. However, it is not marked with a “D” revision mark.

To display/hide revision marks while editing programs offline

Step 1. From the Tools menu, choose Options. A tabbed dialog box is displayed.

Step 2. From the General tab, locate the Miscellaneous group box.

Step 3. Choose the Show Revisions While Offline Editing option. This setting applies to the current and subsequent Editor sessions until you change the setting again.

Step 4. Click OK to accept the new setting.

1.10 About Program Properties

The Program Properties dialog box contains three tabs:

Use this tab:	For:
Program Info	viewing information about the program
Scan Info	defining how the program should be scanned and the program's scan time.
Error Log	viewing the online error log

To access Program Properties from the File menu

Step 1. Make sure no rung or instruction is selected.

Step 2. From the File menu, choose Properties. The Program Properties dialog box is displayed.

To access Program Properties from the pop-up menu

Step 1. Make sure no rung or instruction is selected, and position the mouse in the program's grid area away from any instruction or wire.

Step 2. Press the right mouse button. A pop-up menu is displayed.

Step 3. From the pop-up menu, choose Properties. The Program Properties dialog box is displayed.

See the AutoMax Enhanced Ladder Language Reference manual, J2-3094, for how to interpret the entries in the error log. See section 1.11 below for defining program scan time.

1.11 About Program Execution

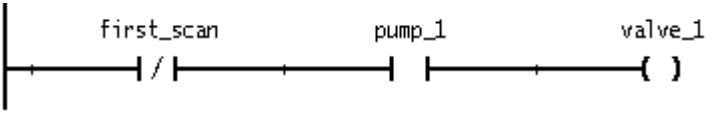
Programs can execute by being scanned or event-driven. An event is a flag or indicator that one program can set or raise and another program can wait for.

Scanned programs execute based on the scan time in ticks you define. For a ladder program, you can define how often you want the program to execute by defining scan time in ticks. The actual scan time of a program is determined by this formula:

*scan time (ms) = tick rate of the Processor module * number of ticks.*

You define the tick rate for the Processor module when you add it to your rack configuration. By changing the tick rate, you can change the time base for program execution. This allows you to run a program based on a unit of your choice. You can set the tick rate in increments of 0.5 ms between 0.5 ms and 10.0 ms. The default value is 5.5 ms. The tick rate is defined separately for each Processor in a rack. For more information, see section 1.11.1.

Event-driven programs can be controlled by either hardware events or software events. When an event-driven is first started (run), the program executes once to completion. After this initial execution, the program then waits for the designated event to occur. If you have program logic that you do not want to run during the initial execution of the program, use the pre-defined *first_scan* variable. An example is shown here:



Hardware events are generated by an external condition. These AutoMax modules can set a hardware event:

- UDC module (57652)
- Pulse Tach Input module (57C421)
- Resolver Input module (57C411)
- 2-Channel Analog Input module (57C409)
- 32-Bit Digital Input module (57C419)

However, hardware events cannot be used on the AutoMax PC3000. See a module's instruction manual for information about interrupts. For more information, see section 1.11.2.

Software events are flags generated by other programs. Software events can be set by BASIC, control block, or ladder programs. For information about programming software events, see your programming language's instruction manual and section 1.11.3.

1.11.1 Specifying a Scan Time To Control Program Execution

For a ladder program, you can define how often you want the program to execute by defining scan time in ticks. The actual scan time of a program is determined by this formula:

*scan time (ms) = tick rate of the Processor module * number of ticks*

Define the scan time by using the Scan Info tab under Program Properties.

To define how often you want your program to execute

Step 1. Access Program Properties. Make sure no rung or instruction is selected, and choose Properties from the File menu.

Step 2. Choose the Scan Info tab.

Step 3. Select Scanned for the Scan Mode.

Step 4. In the Scan Time (ticks) field, enter the number of ticks you want to use. Enter a value from 1 to 32767. The default value is 20.

For example, if you want a program to execute every 30 ms, and the Processor has a tick rate of 3 ms, enter 10 in the Scan Time field.

Step 5. Click OK to accept the changes.

1.11.2 Specifying a Hardware Event To Control Program Execution

Hardware events are generated by an external condition, such as a digital input or a programmed hardware event from a Resolver module. Specify the hardware event that triggers a program's execution by using the Scan Info tab under Program Properties.

You can specify a timeout for the hardware event. This lets you specify the maximum amount of time in ticks that can pass before the hardware event occurs. You can use a timeout as a safeguard in case something happens to the module you are using to generate the hardware event. If the event does not occur before the timeout time, a rack STOP ALL error occurs.

To specify a hardware event

Step 1. Access Program Properties. Make sure no rung or instruction is selected, and choose Properties from the File menu.

Step 2. Choose the Scan Info tab.

Step 3. Select Event Driven for the Scan Mode.

Step 4. Select Hardware as the Event Type.

Step 5. In the Name field, type the name of the hardware event that will trigger the program's execution.

Step 6. In the ISCR Variable Name field, type the name of the variable associated with the ISCR (Interrupt status and control) register on the hardware module being used to generate the interrupt.

Step 7. Do one of the following:

To:	Do this:
specify a timeout for the hardware event	a. Select the Timeout Enabled b. Enter an integer value between 1 and 32767 You should set the timeout to a time longer than what you expect the actual time will be. A good general rule is 1.5 times longer than the expected time between events.
not use a timeout	Make sure the Timeout Enabled checkbox is not selected.

8. Click OK to accept the changes.

1.11.3 Specifying a Software Event To Control Program Execution

A software event is a flag set by another program. Specify the software event that triggers a program's execution by using the Scan Info tab under Program Properties.

To specify a software event

Step 1. Access Program Properties. Make sure no rung or instruction is selected, and choose Properties from the File menu.

Step 2. Choose the Scan Info tab.

Step 3. Select Event Driven for the Scan Mode.

Step 4. Select Software as the Event Type.

Step 5. In the Name field, type the name of the software event that will trigger the program's execution.

Step 6. Click OK to accept the changes.

1.12 Running and Stopping Ladder Programs

Use the Online Task Manager to place a program into run or to stop a program. If you are editing online, you must pause the Editor before you can access the Online Task Manager, since they both share the same communication channel. See "Pausing the Editor," section 5.3, for more information.

1.13 Exiting the Ladder Editor

Exit the Editor with the Exit command. The Exit command closes all program windows and closes the Editor.

The Editor prompts you to save any changes to open programs before exiting. If you choose to close an online program that has changes that have not been accepted, the Editor prompts you to return to the editing session or close. Selecting close discards all the changes.

To exit the Editor

Step 1. From the File menu, choose Exit.

Step 2. If you have made changes to any open ladder program since you last saved it, choose one of the following buttons in the message box that asks if you want to save the changes:

To:	Click:
keep the changes	Yes
not save the changes	No
not close the program	Cancel

Tip

You can also exit the Editor by choosing Close from the Editor Control menu.

2.0 Editing Programs

This chapter describes how to use the Editor to edit programs offline. See chapter 5 for information about editing programs online.

2.1 Starting a Rung

The first step in entering ladder logic is inserting the first instruction in the rung. Place the first instruction of a rung against the left power rail.

All rungs are aligned against the left power rail.

To start a rung

Step 1. Click on the Ladder Language toolbar button that accesses the group of instructions you need.

An instruction palette appears.

Step 2. Point to the instruction you want to insert, and press and hold the left mouse button.

Step 3. While holding down the mouse button, drag the instruction into the program area and next to the left power rail. While you are dragging the instruction, the cursor changes to indicate whether you can drop the instruction and where it will be placed. Instructions destined for the left power rail will look like  or 

Step 4. When the instruction is where you want to place it, release the left mouse button, dropping the instruction into place.

IMPORTANT

You must place the first instruction of a rung so that it attaches to the left power rail to start a rung. If you place it anywhere else in the program window, you must then cut, paste, or drag to connect it to the power rail. You cannot draw a wire to connect it to the left power rail.

Tip

If you frequently use a particular instruction on a toolbar palette, you may want to turn the toolbar palette on. Once a toolbar palette is on, just point to the instruction you want to insert, and press and hold the left mouse button to drag the instruction from the palette.

2.1.1 Defining the Horizontal and Vertical Grid Limits Per Rung

You can define limits for the size of the grid used by the Editor to lay out each rung. This lets you make sure that your program printouts fit on your preferred page size without the rungs being split up.

Relay instructions occupy a grid space of 1 x 1. More complex instructions occupy at least 3 units horizontally by 2 to 8 units vertically.

The default grid size per rung is 40 horizontal units by 24 vertical units, which is also the maximum grid size per rung.

The Editor lets you place instructions outside the grid limits. Rungs that exceed either limit cause a warning to be displayed during a verify operation. The limits you set are checked only when you perform a verify operation.

To define the horizontal and vertical grid limits per rung

Step 1. From the Tools menu, choose Options. A tabbed dialog box is displayed.

Step 2. From the General tab, locate the Per Rung Instruction Grid group box.

Step 3. Enter the desired values for the Horizontal and Vertical fields. The Editor uses these new values for all subsequent programs you open.

Step 4. Click OK to accept the new grid size.

2.1.2 Turning the Grid On and Off

Use the grid to help you see where instructions will be placed and where you can draw wires. You can select whether to display the grid used by the Editor to lay out rungs. The grid is displayed using + symbols.

To turn the grid on or off from the View menu

- From the View menu, choose Grid.

2.2 Selecting Rungs

Before you can copy, cut, or move a rung, you must first select it. You can select a rung by doing any of the following:

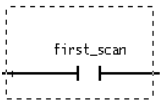
- Click on or near the rung number.

  or

- Click on the rung description text (when displayed).

  or

- Draw a selection box around the entire rung. A selection box is a dotted-line rectangle like this:



A selected rung looks like this: 

Tip

If rung numbers are not displayed, you can still select the rung by clicking in the rung status area next to the rung.

To select a rung by drawing a selection box

Step 1. Think of the rung you want as being enclosed in a box, and position the pointer on the white space at either the top or bottom "corners."

Step 2. Press the left mouse button. The cursor changes to +.

Step 3. Draw a selection box around the rung by moving the mouse diagonally across the rung. You cannot draw a selection box to encompass more than one rung.

Tip

If the rung you are selecting extends beyond the screen display, you can scroll the screen to display the remaining rung logic by extending the selection box against any side of the program window.

Step 4. Once the rung is enclosed in the box, release the mouse button. The rung is now selected.

To select multiple, contiguous rungs by dragging the mouse

Press and hold the mouse button in the rung status area, and drag the mouse in either direction (up or down) to select additional rungs.

To select multiple, contiguous rungs by selecting the first and last rung

Step 1. Select a rung.

Step 2. Scroll the screen as necessary to display the last rung you want to select.

Step 3. Hold down SHIFT and select the last rung by clicking in the rung status area next to the rung.

To select all the rungs in a program

- From the Edit menu, choose Select All

  or

- Press CTRL+A

  or

- From the program window pop-up menu, choose Select All.

2.3 Entering Rung Descriptions

Rung descriptions help you document the program's logic so that others can understand the function of each rung and help make troubleshooting easier. Rung descriptions can contain a maximum of 16 lines, with each line containing a maximum of 80 characters of text.

You must turn on rung descriptions to view them in the program.

To enter a rung description in the program window

Step 1. Make sure Rung Descriptions are turned on.

Step 2. Select the rung to which you want to add or edit a description. The rung description field, which is a dotted line box, is displayed above the rung. For more information, see section 2.2.

Step 3. Click in the rung description field and enter the description.

Tips

- Start a new line by pressing CTRL+ENTER.
- Move around in the rung description field by using the mouse, HOME, END and the arrow keys. Delete text by using BACKSPACE or DELETE.
- You can also cut, copy, and paste text in rung descriptions. See the AutoMax Ladder Editor and Enhanced Ladder Editor help file for more information.

To enter a rung description in the Rung Properties dialog box

Step 1. Select the rung to which you want to add or edit a rung description. For more information, see section 2.2.

Step 2. From the File menu, choose Properties.

Step 3. In the Rung Description field, type in the description.

Step 4. Click OK to add the description.

Tip

You can also access the Rung Properties dialog box from the pop-up menu. Select the rung or point to it. Then, click the right mouse button and choose Properties from the pop-up menu.

2.4 Inserting Instructions into a Rung

Build a rung by using the mouse to drag and drop instructions from an instruction palette and connect them together by drawing wires. The program area is like a blank canvas—you drag instructions from the instruction palettes and drop them in the program area. But first, you must start a rung by dragging an instruction from an instruction palette and dropping it very near the left power rail. To help make inserting instructions easier, the insert cursor indicates when and in which direction you can drop an instruction. See section 2.1 for more information on starting a rung.

When dropped into the rung, the instructions indicate where you can attach another instruction or wire. Contact instructions have connections on either side, but coil instructions can only connect to another instruction on the left side.

For instruction blocks like ADD, you can connect other instructions to the longer lines. These longer lines attach to Boolean input and output parameters.

An inserted instruction belongs to the rung above it if the newly inserted instruction does not start a rung.

To insert instructions into a rung

Step 1. Click on the Ladder Language toolbar button that accesses the instruction palette you want to use.

The instruction palette appears.

Step 2. Point to the instruction you want to insert.

Step 3. Press and hold the left mouse button while dragging the instruction into the program area.

Step 4. When the pointer is where you want the upper-left corner of the instruction to be, release the left mouse button, dropping the instruction into place.

If you drop an instruction very near another, the newly-placed instruction automatically connects to the one adjacent to it. Similarly, you can insert an instruction between two instructions by simply dragging it between the existing two.

Tip

If you frequently use the instruction on a toolbar palette, you may want to turn the toolbar palette on. See “Turning the Toolbars and Palettes On and Off” in Appendix A.

Tip

If you delete or cut the first instruction of a rung, the left power rail connection stays, indicating that you can insert a new instruction. Insert the new instruction to the right of the left power rail connector.

Tip

You can move or copy one or more instructions from one rung and insert them into another rung. See “Moving Instructions” and “Copying Instructions” in the AutoMAx Ladder Editor online help.

2.5 Connecting Instructions (Drawing Wires)

Connect instructions when you want to create branches (parallel logic) or to connect two or more instructions together that were placed too far apart for them to connect automatically. Connect instructions by drawing wires.

You can use the grid markers to help you determine where instructions will be placed and where you can draw wires.

To connect instructions

Step 1. Place the cursor near a connection point for an instruction. For relay logic, the connection point is either the left side (for coils) or both sides (for contacts). For block instructions, the connection points are the free ends of the long lines.

When the cursor is near a connection point, it changes to include a pencil: 

Step 2. Press and hold the left mouse button. The cursor changes to a pencil.

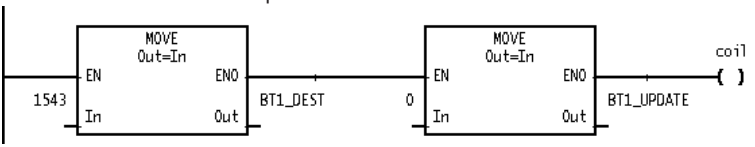
Step 3. Draw the wire by moving the mouse towards the instruction to which you want to connect. You can draw a continuous wire to connect instructions in a branch and the coils at the end of the rung. A dotted line indicates where the wire will be placed.

Step 4. When you reach the destination connection point, release the mouse button.

The wire is drawn to connect the instructions.

2.6 Connecting Multiple Instructions by Using the ENO Output Bit

Quickly build a rung by connecting a relay instruction or a Boolean input of an instruction to the ENO output parameter of an instruction block. For example:



The Boolean input parameter's or relay instruction's state is linked to that of the ENO output parameter.

The ENO output parameter for an instruction follows the state of that instruction's EN input parameter unless an error occurs. When an error occurs within an instruction, the ENO output parameter is set according to the pre-defined variable ERROR_ENO. Because the default value of ERROR_ENO is false, instructions connected to the ENO output are disabled when an error occurs in the instruction block containing the ENO output. Should you want to continue evaluating and executing the logic connected to an ENO output, set the variable ERROR_ENO true.

2.7 Selecting Instructions

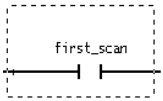
Before you can copy, cut, clear, or move an instruction, you must first select it.

To select an instruction

- Click on any part of the instruction such as the instruction itself, a variable used in the instruction, or the description.

  or

- Draw a selection box around the instruction. A selection box is a dotted-line rectangle like this:



A selected instruction looks like this: 

To select an instruction by drawing a selection box

Step 1. Think of the instruction you want to select as being enclosed in a box and position the pointer on the white space at either the top or bottom “corners.”

Step 2. Press the left mouse button. The cursor changes to +.

Step 3. Draw a selection box around the instruction by moving the mouse diagonally across the instruction. Once the instruction is enclosed in the box, release the mouse button. The instruction is now selected.

To select multiple, contiguous instructions by drawing a selection box

- Draw a selection box to encompass the instructions you want to select. You can only select instructions that belong to the same rung.

  or

Step 1. Select an instruction.

Step 2. While pressing SHIFT, place the mouse pointer on the selected instruction and press the left mouse button.

A selection box appears.

Step 3. While holding down the mouse button, drag the selection box to encompass the instructions that you want to select.

Tip

If the instructions you are selecting extend beyond where they are displayed on the screen, you can scroll the screen to display the remaining instructions by extending the selection box against any side of the program window.

2.8 About Instruction Properties

The Instruction Properties dialog box contains two tabs:

Use this tab:	For:
Instruction Info	• changing the instruction's type • viewing the variable names used in the instruction's input and output parameters
Variables	• viewing the variables used in the instruction • entering a variable description • changing variable's data type, scope, initialization, and display format • specifying a maximum array index

To access Instruction Properties from the File menu

Step 1. Select an instruction.

Step 2. From the File menu, choose Properties. The Instruction Properties dialog box is displayed.

To access Instruction Properties from the pop-up menu

Step 1. Point to the instruction, and press the right mouse button.

Step 2. From the pop-up menu, choose Properties. The Instruction Properties dialog box is displayed.

To display the variable properties tab, choose Variables. To display information about the instructions, choose Instruction Info. Quickly toggle between the two tabs by using CTRL+TAB.

2.9 Changing an Instruction Type

You can change an instruction to another compatible instruction without having to re-enter the variable names for the parameters. This feature helps reduce the time required to make logic changes.

Compatible instructions are those with the same basic function and/or the same number of inputs and outputs. For example, you can change an NOI instruction to a NCI instruction because these instructions are compatible relay input instructions. Conversely, you cannot change an NOI instruction to a CO or an ADD instruction because these instructions are not compatible.

To change an instruction type by using Find and Replace

Step 1. Click on 

The Find dialog box is displayed.

Step 2. In the Objects group box, choose Instruction.

Step 3. In the Find Instruction list box, choose the instruction for which you want to search. This list only displays those instructions that are compatible with the selected instruction.

Step 4. Choose Find.

The Editor searches the program for the instruction you specified.

Step 5. Choose the Replace button.

Step 6. In the Replace Instruction list box, choose the instruction that you want to use in place of the instruction you are searching for. This list only displays those instructions that are compatible with the selected instruction.

Step 7. Choose Replace to replace the current instruction with the new one or Replace All to replace all occurrences of an instruction with another.

To change an instruction type by using its Instruction Properties

Step 1. Select the instruction you want to change.

Step 2. From the File menu, choose Properties.

Step 3. Choose the Instruction Info tab.

Step 4. From the Type list box, choose the instruction type that you want to use in place of the selected instruction. This list only displays those instructions that are compatible with the selected instruction.

Step 5. Click OK to accept the change.

Tip

You can also access the Instruction Info tab from the pop-up menu. Point to the instruction that you want to change, press the right mouse button, and choose Properties from the pop-up menu.

Tip

You can also change the information about the variables used in the instruction. Simply choose the Variables tab or press CTRL+TAB to display the Variable Properties.

Tip

To select the instruction and access the pop-up menu, place the mouse pointer over the instruction and click the right mouse button.

2.10 Assigning Variables and Constants to Ladder Instruction Parameters

Each instruction includes at least one input or output parameter. The more complex instructions contain parameters for multiple non-Boolean inputs and outputs. Each parameter has a variable name field in which you enter the variable name or constant that you want to assign to the parameter.

To help make assigning variables to instruction parameters easier and faster, variables are automatically assigned a default data type and scope when they are first entered. The default type is that most likely to be used by the instruction. For example, the default type for a variable name entered for a relay instruction is Boolean. But for a JMP, the default type is label. For most block instructions, the default type for input and output parameters is integer.

When you enter an element-indexed variable, a default maximum array index is automatically assigned, which you can later change in the Variable Properties.

The scope of the variable is determined by the case of the first letter of the variable name you type. If the letter is upper case, the variable defaults to being a global variable. If the letter is lower case, the variable defaults to being a local variable.

To assign variables or constants to ladder instruction parameters

Step 1. Select the instruction for which you want to assign variables. The variable name field appears as a dotted-line box.

Step 2. Click in this dotted-line box. The outline becomes solid and a vertical text cursor appears.

Step 3. Type in the variable name or a constant you want to use following the naming conventions of the Editor. See the reference information for the instruction you are programming for more information about the allowable variable types. The scope of a variable can be either global or local. When you first enter a variable, its scope is defined based on the case of the first letter you type. An upper case letter defines the variable to be global. A lower case letter defines it to be local. See section 2.15 for more information about a variable's scope.

Tip

If the variable has not been used before, a default data type and scope is automatically assigned. If the variable is an array, the maximum array index defaults to the value you entered as the element of the variable. For example, if you entered part[5] the value of 5 would be entered into the Maximum Array Index field of the Variable Properties.

If the element-index of an array is a variable, the Maximum Array Index field contains a value of 1.

Tip

If the variable has been used before, you can save typing by using Variable Smart-Matching. See section 2.12 for more information about Variable Smart-Matching.

Tip

To enter hexadecimal constants:

- End hexadecimal constants with an “h” or “H”.
  For example: 607H
- If the hexadecimal constant begins with a letter, enter a leading 0 before the value.
  For example: 0A607H
- If the most significant bit of the hexadecimal constant is set and the constant is fewer than 8 digits, sign-extend it with “F’s” into an 8-digit hexadecimal value.
  For example: Enter 9C40H as 0FFFF9C40H

The Editor converts a hexadecimal value to a decimal constant, except for the logical and Masked Move instructions.

2.11 Entering Variable Descriptions

Variable descriptions help you document variables used in instructions. Each variable can have its own description; however, the description applies to simple or array variables and not to indexed variable names. For example, the description displayed for the variable TANK.fill would be for the variable TANK.

The descriptions can contain a maximum of 40 characters.

You must turn on variable descriptions to view them.

To enter a variable description by clicking into the variable description field above a variable

Step 1. Make sure that the variable descriptions are turned on.

Step 2. Select the instruction.

Step 3. Click in the variable description field above the variable for which you want to enter a description. When selected, the variable description field is outlined by a solid border.

Step 4. Enter a description containing a maximum of 40 characters.

Tip

A shortcut to entering a variable description is to click twice on the variable name and press TAB to advance to the variable description field. Enter a maximum of 40 characters of text.

To enter a variable description by using the Variables tab in the Instruction Properties dialog box

Step 1. Select the instruction that contains the variables you want to document.

Step 2. From the File menu, choose Properties. The Instruction Property dialog box is displayed.

Step 3. Choose the Variables tab. The Variables Properties box is displayed.

Step 4. In the Name list box, choose the variable you want to document.

Step 5. Click on the Description field, and type in the description.

Step 6. Click OK to add the description.

Tip

You can also access the Variable Properties dialog box from the pop-up menu. Point to the instruction containing the variables to which you want to add or edit a description, press the right mouse button, and choose Properties from the pop-up menu.

Tip

Move around in the name field by using the mouse, HOME, END, and arrow keys. Delete text by using BACKSPACE or DELETE.

Tip

To select the instruction and access the pop-up menu, place the mouse pointer over the instruction and click the right mouse button.

2.12 Using Variable Smart-Matching

Variable smart-matching is an option you can use to help you enter variable names in less time. The Editor maintains an alphabetical and case-sensitive list of variables you have used. When this option is on and you begin entering letters in the variable name field, the first variable name in the Editor's list that matches the first letter or letters you have typed appears in the variable name field. For example, if you have previously used the variables "pump" and "packet" and you type the letter "p" in a variable name, the word "packet" is inserted into the variable name field. If "packet" is the variable name you want, you can stop typing. If you had typed a "P," global variables (which begin with an uppercase P) would have been offered by the Variable Smart-Matching option.

If variable smart-matching displays a variable name that you only want to use part of, you must delete the extra characters by pressing BACKSPACE or DELETE.

To turn the Variable Smart-Matching option on and off

Step 1. From the Tools menu, choose Options. The Option dialog box is displayed.

Step 2. Choose the General tab and locate the Miscellaneous group box.

Step 3. Choose the Variable Smart-Matching option. This setting applies to all subsequent editing sessions.

Step 4. Click OK to accept the new setting.

To scroll through the available variable names that match the character(s) you have typed

- To advance, press CTRL and the down arrow.
- To move backward through the matching variable names, press CTRL and the up arrow.

The Variable Smart-Matching option provides matches to each part of an element-indexed and/or bit-indexed variable. Enter the element-index or bit-index delimiter and the first letter or letters of the element or bit name. Then, scroll through the variable name choices offered. The choices offered by the variable smart-matching option are determined by how many letters you type in.

Tip

- To append onto a variable name picked from the variable smart-match choices, make sure the variable name is not selected before continuing to type.
- To re-initialize smart-matching, delete characters (by using BACKSPACE), and re-type one or more.

Tip

If you type a question mark (?) as the first letter of a variable name, the Editor automatically smart-matches on the first variable in your variable list.

2.13 Selecting Variable Names

Before you can copy, cut, or clear a variable name you must first select it.

To select a variable

Step 1. Select the instruction that contains the variable you want to edit. The variable name fields appear as dotted-line boxes.

Step 2. Click the dotted-line box. The outline becomes a solid box and the text is selected.

2.14 Changing a Variable's Data Type

When you enter a variable name in an instruction's input or output parameter, a default data type is assigned based on the instruction type. If you want to change a variable's data type to something other than the default, you can change it by using the Variable Properties tab.

IMPORTANT

Changing the properties of a variable affects the variable, not just a particular instance in which it is used. For example, if you change the data type of a variable from timer to integer and that variable happens to be used in a TON instruction, this integer variable will no longer be allowed within the instruction.

To change the data type for a variable

Step 1. Select the instruction containing the variable whose data type you want to change.

Step 2. From the File menu or the pop-up menu, choose Properties. The Instruction Properties dialog box is displayed.

Step 3. Choose the Variables tab. The Variable Properties tab is displayed.

Step 4. If the name of the variable you want is not displayed in the Name field, use the list box to scroll through a list and choose it. The Variable Properties dialog box for that variable is displayed.

Step 5. Using the Type list box, choose the data type for the variable.

Step 6. Click OK to accept the change.

Tip

Integer variables default to a 16-bit integer. If you want the variable to be a double integer (32-bits), you must change its type to double integer.

Tip

Quickly toggle between the Instruction Info and Variables tabs by using CTRL+TAB.

Tip

To select the instruction and access the pop-up menu, place the mouse pointer over the instruction and click the right mouse button.

2.15 Changing a Variable's Scope

The scope of a variable can be either global or local. When you first enter a variable, its scope is defined based on the case of the first letter you type. An upper case letter defines the variable to be global. A lower case letter defines it to be local. You can change a variable's scope by using the Variable Properties tab.

Compound variables (element-indexed or bit-indexed variable names) are treated as separate variables with their own properties, including scope. For example: TANK.fill is a compound variable. *TANK* is a global variable and *fill* is a local variable. For element-indexed and bit-indexed variables using a number as the index, the scope of the variable is determined by the named variable. For example, the scope of the variable TANK.31 would would be determined by TANK, which would appear in the variable list. If the variable were TANK.31, only TANK would appear in the variable list and have a scope.

IMPORTANT

Changing the properties of a variable affects the variable and not just a particular instance in which it is used. If you change a variable's scope, the change applies to every instance where that variable is used.

To change a variable's scope

Step 1. Select the instruction containing the variable whose scope you want to change.

Step 2. Access its Instruction Properties by choosing Properties from either the File menu or the pop-up menu. The Instruction Properties dialog box is displayed.

Step 3. Choose the Variables tab. The Variable Properties tab is displayed.

Step 4. If the name of the variable you want is not displayed in the Name field, use the list box to scroll through and choose it. The Variable Properties tab for that variable is displayed.

Step 5. Click on either the Global or the Local option. The variable name changes to be either all upper case or lower case, depending on which option you chose.

Step 6. Click OK to accept the change.

Tip

Quickly toggle between the Instruction Info and Variables tabs by using CTRL+TAB.

Tip

To select the instruction and access the pop-up menu, place the mouse pointer over the instruction and click the right mouse button.

3.0 Printing Programs

Print a program to mark up or maintain a paper record of it. You can print:

- an entire program or a range of rungs
- rung descriptions
- only rung descriptions
- variable descriptions
- instruction cross-reference
- program cross-reference
- multiple copies of a program

The ladder diagram prints out on the default printer you have defined for the computer. Make sure that this printer is connected and the proper print driver is installed. See the Windows 95 documentation for more information.

You can print a program by choosing:

- the Print button from the Standard toolbar
- Print from the File menu

To print a program from the Standard toolbar

Step 1. Make sure that the program you want to print is in the active program window.

Step 2. Choose 

To print a program using the Print command

Step 1. Make sure that the program you want to print is in the active program window.

Step 2. Do one of the following:

- From the File menu, choose Print.

  or

- Press CTRL+P

  The Print dialog box appears. You can use the Print dialog box to change the number of pages, page orientation, or set new print options.

Step 3. Use this table to help you with your next steps:

To:	Do the following:
print all the rungs in the program	In the Print Range box, select All.
print only the rungs within a selected range	In the Print Range box, select Selected Rungs.
print more than one copy of the program	Enter or select the number of copies using Copies.
define page margins	Choose Page Setup.
define other print options	Choose Options.

Step 4. When you are ready to print the program, choose OK.

The document is sent to the printer that is displayed at the top of the dialog box (the default printer).

3.1 What Is Included in a Printed Copy of a Program

Your program printout includes the items you have chosen to print using the Print Options tab.

A program printout includes:

- header information, which describes the program's path, when it was last edited, and whether the page is the program's source code or cross-reference
- rungs, which you can select to print using the Print Options tab
- variable and rung descriptions, both of which you can select to print using the Print Options tab
- program and instruction cross-references, both of which you can select to print using the Print Options tab; see below for more information on the program cross-reference
- initial value table for the variables that have a user-specified or retained initial value that is different from the default value
- program properties table, which lists pertinent data about the program such as:
  - the program's name and path
  - program's scan time displayed in the Scan Info Tab of the Program Properties
  - the number of rungs and symbols in the program
  - the date the program was created and the date it was last revised
  - the size of the program's image, its verify status, and the amount of online reserve memory it requires; see Appendix D for more information

When you print the Program (task) cross-reference, it prints at the end of the printout and is printed for the entire program, even if you selected only a range of rungs to print. The program cross-reference contains the following items:

variable name table	This table includes each variable used in the task, its data type, the instruction and rung where it is used, its I/O type, the hardware address, its description, and the line cross-reference. If the database does not exist, the text “Unknown” is printed for the descriptions and cross-reference information.
rung table	This table includes information about each variable used as an output in the program, including the rung number where the variable appears, its name, the instruction symbol or mnemonic in which the variable is used, I/O description, and the variable description from the program variable table.
messages	The Editor prints any messages about the cross-reference summary, verify result summaries, print processing, and database processing after the rung table. When printing the program from the Editor, verify warnings are printed if you have not selected Ignore Warnings in the General Tab.

The items in the program are printed in the following order:

1. rungs
2. instruction cross-reference (if selected)
3. initial value table
4. program cross-reference, including the variable table, rung list, legends, scan information, verify error statement, and summary (if selected)

3.2 Setting Print Options

You can choose the type of information to print on your programs' printout. Choose to print or not to print the following:

To print:	Choose:	Result:
a program's ladder logic rungs	Rungs	The rungs are printed.
any description assigned to rung	Rung Descriptions	The rung descriptions print above the rung. The line length is the same as the page width. The description is truncated if it is longer than the page width.
only rung descriptions	Rung Descriptions Only	The rungs, variable descriptions, and instruction cross-reference are not printed. You can also choose to print the program cross-references when you choose Rung Descriptions Only.
any description assigned to a variable	Variable Descriptions	The variable descriptions are taken from program symbol table and are printed above variable name, just as they are displayed in the program.
information that helps you track where variable names are written and where they are read	Instruction Cross-Reference	The instruction cross-reference is printed with each rung. When only a range of rungs is selected, the instruction cross-reference still includes references for the entire program.
variable name table and a rung table	Program Cross-Reference	The program (task) cross-reference prints at the end of the program printout after the rungs. Included is a variable name table and a rung number table. When only a range of rungs is selected, the program cross-reference is still printed and provides references for the entire program.

Choose any print options on the Print Options Tab, which you can access from the Tools menu or from the Print dialog box. These items are printed by default:

- rungs
- variable and rung descriptions
- program and instruction cross-references

To define the print options

Step 1. Access the Print Option Tab by choosing:

- Options from the Tools menu and then the Print Tab

or

- the Options button on the Print dialog box

Step 2. Choose the items you wish to print when you print a program. Your choices are listed within the Print What group box.

Step 3. Click OK.

The options you choose apply to any program that you print.

Tip

To return to using the default print options, select the Use Defaults button.

3.3 Defining the Page Setup

You can change the page margins for the paper copies of the programs that you print.

The default margins are:

- top and bottom margins, 1 inch
- left and right margins, 0.5 inches.

Once you set the page margins, they apply to all subsequent programs that you print.

To set the page margins for the programs that you print

Step 1. Do one of the following:

- Click on 

or

- From the File menu, choose Print

or

- Press CTRL+P

Step 2. At the Print dialog box, choose Page Setup.

Step 3. Enter the new measurements for the top, bottom, left and right margins.

Note: *If you choose margins that are less than the printable area on the printer/paper, the printout is still printed, but the measured margin may not accurately match your selections.*

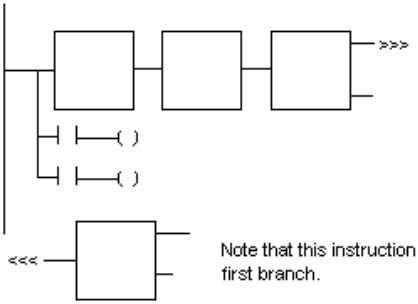
Step 4. Click OK.

3.4 How a Rung Is Printed To Fit on a Single Page

The Editor prints a rung (including rung descriptions, variable names and descriptions, and instruction cross-references) on one page. If a rung or an instruction block cannot be printed entirely on one page and there are other rungs on the page, the Editor forces the logic to a new page. If the rung or an instruction cannot be printed entirely on the new page, the Editor breaks up the logic as follows:

If a rung or instruction cannot be printed to fit on the page:	The Editor:	The split is indicated as follows:
vertically	finishes printing it on a new page	V V V This symbol prints at the bottom of the page, indicating lines continue on the next page. ^ ^ ^ This symbol prints at the top of the page, indicating lines continue from the previous page.
horizontally	wraps it to the next line Wrapped rungs are printed across the page from top to bottom then left to right. This means that if one branch of a rung cannot be printed across the page, the continuation of the branch appears below the last branch of the rung as illustrated in the figure below. When a wrapped rung is printed on a page, the next rung in the program prints on a new page.	> > > This symbol prints on the right side of the page, indicating that instructions are being extended to the next line. < < < This symbol prints on the left side of the page, indicating that the instructions are continuations of the rung or branch above.

Wires and instructions can also be split. See the following example:



3.5 Inserting and Deleting Page Breaks

Use a page break to paginate a program's printout for a cleaner look.

To insert or delete a page break

Step 1. If you are inserting a page break, select a program object with which you want to start a new page. If you are deleting a page break, select a program object that starts a new page.

Step 2. From the Insert menu, choose Page Break.

The Editor inserts or deletes the page break above the selected program object. Page breaks are displayed horizontally across the screen.

3.6 Printing a Range of Rungs

If you do not want to print an entire program, you can print a range of rungs that you have selected.

To print a range of rungs

Step 1. Within the program, select the rungs you want to print.

Step 2. Do one of the following:

- Click on 

or

- From the File menu, choose Print

or

- Press CTRL+P

Step 3. In the Print dialog box, choose Selected Rungs.

Step 4. Define page margins or print options, if necessary.

Step 5. Click OK.

The document is sent to the printer that is displayed at the top of the dialog box (the default printer). If you have selected to print a program cross-reference, one is printed for the entire program.

3.7 Displaying and Printing Coils as Right-Justified

Use the Right Justify Coils after Verify feature to automatically move coils to the right side of the printout so that you can more easily read your program. If coils are not blocked by another instruction, the Editor aligns them against the first vertical page break. Coils that already extend past this page break are moved left and up. (They are not moved to the next page break.)

To display and print programs with right-justified coils

Step 1. From the Tools menu, choose Options.

Step 2. On the General Tab, choose Right Justify Coils after Verify.

Step 3. Click OK.

Coils are only moved out to the grid limit when the limit is to the left of the first vertical page break.

The default option is not to right-justify coils after verifying or printing a program. The rungs are displayed and printed compressed up and to the power rail when you verify or print the program.

3.8 Printing Multiple Copies of Programs

You can print more than one copy of a program at a time.

To print multiple copies of a program

Step 1. Make sure that the program you want to print is in the active program window.

Step 2. Do one of the following:

- Click on 

or

- From the File menu, choose Print

or

- Press CTRL+P

Step 3. Within the Print dialog box, use the Copies spinner box to enter or select the number of copies you want to print.

4.0 Verifying Programs

Use the Verify command to help debug your ladder program. During the verification process, the Editor checks your program and notifies you of any errors that would prevent the program from running. If you choose, the Editor can report less serious issues, called warnings. See section 4.3 for more information. The error and warning messages also include information about the rung and grid location in which the error or warning occurred.

A program whose verify operation results in errors is not successfully verified and cannot be downloaded to a Processor and run. If only warnings result after a verify operation, the program is considered verified. Look at the bottom right corner of a program's status bar or in the Program Info tab (Program Properties) for the program's verification status, either Verified or Not Verified.

As the Editor verifies a program, it automatically compresses the rung to the left and up by deleting rows and columns that do not contain an instruction and minimizing wire lengths. Coils are moved out to the vertical page break if the Right-Justify Coils after Verify option is not selected. See section 3.7 for more information.

The Editor automatically verifies an online program when you choose to commit any online changes or place the program in Test Mode. Online program changes cannot be downloaded to a Processor and run until they are successfully verified.

Each program has its own Verify Output window that lists the error, warning, and status messages generated during the verify process. Because the Verify Output window is separate from the program window, you can easily switch between the program and the verify messages as you troubleshoot your program. The title of the Verify Output window is the name of the program appended by a .LOG extension. If you choose to verify the program again while the Verify Output window is open, the information in the window is overwritten. Within the Verify Output window, you can search for text and print the log file.

When offline programs are verified, the Editor does not check the Variable Configurator or create a verify log file unless you set the applicable options in the General tab of Tool Options. See sections 4.1 and 4.2 for more information.

To verify a program

Step 1. Make sure the program you want to verify is the active program.

Step 2. From the Tools menu, choose Verify.

4.1 Creating a Verify Error Log File

Select the Generate Log File option in the General Options tab to create a log file on disk containing the same error, warning, and status messages as the Verify Output window. This file is stored in the same path as the program with the same name but with a .LOG extension. It can be opened and viewed using a text editor like WordPad®. The Editor generates a log file for each program by default.

To create a verify log file

Step 1. From the Tools menu, choose Options.

Step 2. On the General tab, select the Generate Log File option in the Verify group box.

Step 3. To accept the change, click OK or Apply.

Tip

Deselecting this option prevents the Editor from generating the log file.

4.2 Automatically Checking the Variable Configurator Database While Verifying a Program

Select the Check Variable Configurator Database option in the General tab of Tools Options to automatically check the properties of a variable against the Variable Configurator database. When this option is selected, the Editor determines whether:

- global variables used in the program are declared in the Variable Configurator database
- global variables using Retained Value initialization are declared as non-volatile
- global timer and counter variables are non-volatile
- array variables used in the program have the same dimension as those defined in the Variable Configurator database

The Editor checks the Variable Configurator database by default.

To automatically check the Variable Configurator database while verifying a program

Step 1. From the Tools menu, choose Options.

Step 2. On the General tab, select the Check Variable Configurator Database option in the Verify group box.

Step 3. To accept the change, click OK or Apply.

Tip

Deselecting this option prevents the Editor from checking the Variable Configurator database. This makes verifying a program faster. However, you cannot install and run a program that contains global variables that are not defined in the Variable Configurator database.

4.3 Specifying Whether To Include Warning Messages When Verifying Programs

Select the Ignore Warnings option in the General tab options from the Tools menu to prevent verify warnings from being displayed in the Verify Output window and verify log file (if the log file is generated). Verify error messages are always displayed in the output window and log file. By default, the Editor includes verify warning messages in the output window and log file of offline programs that you verify. The Editor does not include warning messages arising from committing online changes or placing the program in Test Mode.

To prevent warning messages from displaying in the verify log file

Step 1. From the Tools menu, choose Options.

Step 2. On the General tab, deselect the Ignore Warnings option in the Verify group box.

Step 3. To accept the change, click OK or Apply.

4.4 Resolving Verify Errors

The Verify Output window lists error messages and, if you choose, warning messages that help you debug a program. Error messages also include the rung number and grid location of the logic in which the verify error occurred. For example,

This error:

In Rung #8 at location {4,1}

Variables of type Timer or Counter cannot be used on a coil.

Means:

The coil located in the **fourth column** of the **first row** in **rung 8** has a timer or counter associated with it.

Because the Verify Output window is separate from the program window, you can easily switch between the program and the verify messages as you troubleshoot your program.

Tip

To switch between the two windows, press CTRL+F6

5.0 Editing Programs Online

Editing a program online lets you modify ladder logic and the initial values of variables as well as set, force, and unforce variables. You can be connected to a Processor via a serial connection or a PC Link interface connection. If you are connected via the PC Link module, you can have multiple online programs from different networked racks opened simultaneously.

While a program is running, you can view the power flow of executing logic, variable state information, and any instruction runtime error codes generated.

An Overview of Editing and Downloading the Changes

Step 1. Open the program by using the Monitor PC Program command from the Editor, Task Manager, or System Configurator.

Step 2. Make your changes to the online program.

Step 3. Accept the changes.

Step 4. Download the program modifications to the Processor.

About the Changes You Can Make

While editing an online program, you can add new variables with some limitations and restrictions, but you cannot add or edit variable or rung descriptions. See "About the Limitations for Inserting and Modifying Rungs in Online Programs" in Appendix D for more information. When you add global variables, make sure the variables are non-volatile and do not have retained initialization data. You must define global variables in the Variable Configurator database. If you enter a global variable in an online ladder program and that variable is not in the Variable Configurator database, you get a verify error when you commit the online changes.

IMPORTANT

Add global variables to the Variable Configurator database before making the online changes.

You can also view rung and variable descriptions. When opening an online program, you can choose whether or not you want to display rung and variable descriptions. When you open an online program, the logic is uploaded from the Processor to your computer. If you choose to display the variable and rung descriptions, these are extracted from the program's source file, which must be accessible by located on the computer you are using to edit the online version of the program. For more information, see "Choosing To Display Descriptions While Editing/Viewing an Online Ladder Program" in the AutoMax Ladder Editor and Enhanced Ladder Language online help.

About Accepting and Downloading Your Changes

When you set, force, unforce, or change a variable's initial value, the change takes effect immediately in the online program. Other changes such as adding, deleting, or modifying a rung must be accepted before the change takes effect. You can accept one or all rungs that have been added, deleted, or changed. Accept changes that you want to download to the Processor. Once any changes are accepted, you can temporarily load them into the Processor (place the program in test mode) or permanently download them to the Processor (commit the changes).

Before any changes are downloaded to the Processor, either permanently or temporarily, the changes are verified using the same rules used for verifying offline programs. If the program does not pass the verification process, you must correct the errors and accept them again. An edited online program is downloaded to a Processor only when all the accepted changes to the program have been successfully verified. The verified changes are downloaded to the Processor as a group, not as individual rungs. See "About How Changes Made to Online Programs are Verified" in section 5.9 for more information.

About Accessing the Online Task Manager

You can access the Online Task Manager from the Editor, Task Manager, or the System Configurator. The Online Task Manager and the Editor cannot share the same communication channel. If you need to use the Online Task Manager while editing an online ladder program, you must place the Editor in a paused state. By pausing the Editor, you free up the communication channel so the Online Task Manager can use it. See section 5.3 for more information.

5.1 About the States of Online Programs

An online program is in one of the following states at any time:

State:	Description:	How the state is indicated:
Run	The program is actively executing in a Processor and power flow is displayed.	The word "RUN" appears the Editor's status bar, and the left power rail is highlighted in the power flow color. In the Online Task Manager, the word "Run" appears in the status column.
Stopped	The program is **not** actively executing.	The word "STOP" appears the Editor's status bar. In the Online Task Manager, the word "Stopped" appears in the status column.

State:	Description:	How the state is indicated:
Test mode	The program is actively executing rungs, but the changes made to an online program are not permanently installed in the Processor. You can test the changes you made to an online program.	The text “Test Mode” appears after the file name in the window’s title bar. See “Testing Programs (Test Mode),” section 5.8, for more information.
Paused	When the Editor is paused, the programs continue to run in the Processor, but their display in the program window is not updated. The Processor connection is freed so you can access the Online Task Manager. Because the Online Task Manager and the Editor cannot share the same communication channel to the Processor, the Pause command allows you to place the Editor into a paused state. The online program windows that are paused after you select the Pause command from the Online menu are: • windows containing your online programs • Original Program window The Set/Force/Unforce dialog box is also closed.	The status bar for windows that are in the paused state is blank and the text “Paused” appears before the file name in the window’s title bar. See “Pausing the Editor,” section 5.3, for more information.

5.2 Monitoring an Online Ladder Program

Before you can edit an online program, you must open it using the Monitor PC Program command from the Online menu. You access this command from the Editor, Task Manager, or System Configurator.

To monitor an online ladder program

Step 1. Close the Online Task Manager application if it is running.

The Online Task Manager uses the same online connection as an opened online program. Both applications cannot be accessed at the same time.

Step 2. Do one of the following:

- From the Online menu of the Editor, Task Manager, or Software Configurator, choose Monitor PC Program.

or

- Click on 

The Monitor PC Program dialog box is displayed.

Step 3. Using the Look In area of the dialog box, navigate to the location containing the ladder program you want to open. You can look in these folders:

- Direct – for programs available in the Processor to which you are connected
- Network – for programs located on Processors connected to the network

Step 4. Once you have located the program you want to monitor, select the program's icon displayed in the Files area of the dialog box. To select more than one program to open, do one of the following:

- Draw a selection box around the programs you want to open.
- To select individual programs, press [CTRL] while selecting the program icons with the mouse.
- To select all programs, select the first program, and press [SHIFT] while selecting the last program.

Notice that the program's name appears in the File name field.

Step 5. If you want to view the rung and variable descriptions while monitoring the online program, see "Choosing To Display Descriptions While Editing/Viewing an Online Program" in online help.

Step 6. Click OK to open the program.

The program you selected is opened.

Tip

You can have multiple online programs opened at once.

5.3 Pausing the Editor

Because the Online Task Manager and online programs in the Editor use the same communication channel, you must place the Editor in the Paused state before you can use the Online Task Manager. The programs continue to run in the Processor, but their display in a program window is not updated.

IMPORTANT

Before pausing the Editor's online program windows, you might want to commit any pending changes. All pending changes are lost once an online program is paused and then removed from the paused state.

Any trigger information is also lost when you remove online program windows from the pause state.

To pause or un-pause the Editor

- From the Online menu, choose Pause.

  or

- Click on 

While online programs are paused, the status bar is blank and the word "Paused" appears in the programs' title bar.

Remove the programs from the paused state when the communication port is again available. The Editor re-establishes communication to all available ports and notifies you when a connection is not available. If a connection is not available, the online programs stay paused.

For more information about the paused state, see section 5.1.

5.4 About Power Flow

Power flow indicates the rungs and instructions that are executing in an online program. It is shown as a bar of color that highlights wires that are connected together. You can define the color of power flow using the Colors Option tab of program properties.

The state of a variable is shown as a solid block of color in the contact. For positive transition contacts (PTI), the block of color is shown when the contact is true, but power flow from the contact is shown only during the scan where the transition is true. The same concept applies to the negative transition contacts (NTI). The block of color in an NTI is shown when the contact is false, but the power flow from the contact is shown only during the scan in which the transition is true.

Hatch marks in the rung status area identify rungs that are not executing. For non-executing rungs, the power flow and last state information you see are those from the last time the rung executed since you started monitoring the online program. Factors causing rungs not to execute are:

- They could be skipped via a Jump instruction.
- The program is executed based on an event.
- The program is stopped.
- The scan time of the program is very long.

5.5 Monitoring Data in an Online Program

While a program is running online, the data is displayed as follows:

- Forced variables are highlighted in the Forced Variable color, defined via the Color Option tab of the Program Properties.
- Numeric values are displayed under the variable name. You can define the data display format using the Variable Properties tab for the variable but only in an offline program.
- A timer's elapsed value is displayed under the timer variable name.
- A counter's current value is displayed under the counter variable name.
- If the preset of a timer or counter variable is changed via logic, the new preset value is displayed under the original preset value.
- For variables using element or bit-indexing, the display format is that of the base variable. For example, if you specified variable *array* to be displayed as decimal and the variable *element* to be displayed in hexadecimal, the value of the variable *array[element]* would be displayed in decimal.
- For element-indexed array variables (array[element]), the value displayed is that of the element.

5.6 Accepting Changes Made to Online Programs

After editing an online program, you must approve the changes before the Editor can download them to the Processor by using the Online menu's Accept command. You can accept modified, added, or deleted rungs either individually or as a group. You can also reject any change. Rejected changes are not recoverable.

You do not have to accept changes to an online program that are made through the Set/Force/Unforce dialog box or that caused the initial value of a variable to be changed because these changes take effect immediately.

To accept or reject changes made to an online program

Step 1. When you are finished editing an online program, do one of the following:

- Choose Accept from the Online menu

  or

- Click on 

The first edited rung is highlighted and displayed along with the Accept Online Changes Dialog box.

Step 2. From the Accept Online Changes dialog box, perform one of the following actions:

To:	Choose this button:	Result:
Approve the displayed rung.	Accept	The next edited rung is displayed for you to accept.
Approve all the rungs that you added, deleted, or modified.	Accept All	The Commit Online Changes dialog box is displayed. You can immediately download the changes to the Processor or temporarily download the changes to the Processor by placing the program in Test Mode. See “Downloading Changes Made to an Online Program (Commit Online Chnges),” section 5.7.
Discard the changes made to the rung.	Reject	The Editor returns the rung to its original construction.
Cancel the Accept process and return to editing the online program.	Cancel	The Accept Online Changes dialog box disappears. You can resume editing the online program. Your current modifications are unaffected.

If you have accepted at least one rung, you can commit the change or place the program in Test Mode. If not, you can resume editing the program.

5.7 Downloading Changes Made to an Online Program (Commit Online Changes)

After accepting added, deleted, or modified rungs, you must commit the changes before the Editor can download them to the Processor. You use the Commit Online Changes dialog box to do this. The Editor automatically displays this dialog box after you have finished accepting the changes made to an online program. You must have accepted at least one online change for the Editor to display the dialog box.

From the Commit Online Changes dialog box, you must choose how to download the online changes to the Processor. You can download the changes to the Processor by sending the changes to the Processor and either:

- immediately install the changes in the program by selecting Commit
- temporarily install the changes in the program by selecting Test Mode

The changes are downloaded to the Processor as a group.

To download changes made to an online program (commit)

Step 1. Accept the changes to an online program. See "Accepting Changes Made to Online Programs," section 5.6.

The Editor displays the Commit Online Changes dialog box.

Step 2. From the Commit Online Changes dialog box, choose one of the following:

To:	Choose this button:	Result:
Send the accepted online changes to the Processor and install them.	Commit	The online changes are verified. See "About How Changes Made to Online Programs are Verified," section 5.9, for more information. Once the changes are successfully verified, they are downloaded to the Processor.
Send the accepted online changes to the Processor and temporarily install them.	Test Mode	The online changes are verified. See "About How Changes Made to Online Programs are Verified," section 5.9, for more information. Once the changes are successfully verified, they are temporarily downloaded to the Processor. The program is placed into Test Mode so that you can see how your changes perform before they are permanently installed in the Processor. See "Testing Programs (Test Mode)," section 5.8, for more information.
Cancel the Commit process and return to editing the online program.	Cancel	The Commit Online Changes dialog box disappears. You can resume editing the online program. Your current modifications are unaffected. When you are finished editing, you must begin the Accept process again.

5.8 Testing Programs (Test Mode)

Selecting Test Mode from the Commit Online Changes dialog box places the program in the Test Mode state, which provides you with an opportunity to make sure that the changes you made work properly before you permanently install them in the Processor. Once a program is in Test Mode, you can:

- commit the online changes, which permanently installs the program in the Processor (see section 5.8.1 for more information)
- take the program out of Test Mode, which allows you to resume editing the program with your current modifications unaffected (see section 5.8.2 for more information)
- cancel all changes

While a program is in Test Mode, you cannot make any changes to it or download a configuration to a rack.

To place an online program in Test Mode

Step 1. After editing an online program, choose Accept from the Online menu.

Step 2. Accept the online changes.

Step 3. From the Commit Online Changes dialog box, choose the Test Mode button.

The online changes are verified. See "About How Changes Made to Online Programs are Verified," section 5.9, for more information. If the changes are successfully verified, they are installed in the Processor. To make the changes permanent you must commit them.

5.8.1 Committing an Online Program in Test Mode (Commit Changes in Test Mode)

Once a program is in Test Mode and you are satisfied with the changes you made, you must commit these changes before they can be installed permanently in the Processor. Use the Commit Test Mode changes command from Online menu.

To commit save an online program in Test Mode

Step 1. Make sure the program is in an active program window.

Step 2. From the Online menu, choose Commit Test Mode Changes. The Editor prompts you to confirm your decision.

Step 3. To commit the changes, click Yes. Clicking No removes the question dialog box, leaving the program unchanged.

Once the rungs are successfully committed, the Editor removes the program from Test Mode and displays the rungs in the non-modified rung color.

5.8.2 Removing an Online Program from Test Mode

If you want to continue making changes to an online program that is in Test Mode, you must first take it out of Test Mode. You can do this by using the Quit Test Mode command from the Online menu. Taking a program out of Test Mode removes the changes from the Processor. The modified rungs are no longer running on the Processor; the original rungs are now executing.

IMPORTANT

If you delete a counter data structure as part of the online changes, the counter has retained its value while the changes were in Test Mode. When you remove the program from Test Mode, the counter is storing a current value since the change that deleted the counter was not accepted.

If you delete a timer data structure as you edit an online program, the timer is reset when you remove the program from Test Mode without accepting the online changes.

To remove an online program from Test Mode while retaining pending changes

Step 1. Make sure the program is in an active program window.

Step 2. Do one of the following:

- From the Online menu, choose Quit Test Mode

  or

- Click on 

  The Editor prompts you to confirm your decision.

Step 3. To remove the program from Test Mode, click Yes. Clicking No removes the question box, leaving your program in Test Mode.

The online program continues to display the modified rungs. You can continue to make changes or cancel all the online changes. You must accept added, deleted, or modified rungs before they can be installed in the Processor again.

5.8.3 Canceling All Changes Made to an Online Program

You can cancel any pending changes made to an online program or remove the program from Test Mode by using the Cancel All Changes command. Pending changes are those that have not yet been committed to the Processor. You cannot cancel changes to a variable that resulted in setting, forcing, or unforcing it or changing its initial value, since these changes take effect immediately.

To cancel all changes made to an online program

Step 1. Make sure the program is in an active program window. The program can be in Test Mode.

Step 2. From the Online menu, choose Cancel All Changes. The Editor prompts you to confirm your decision.

Step 3. To cancel the changes, click Yes. Clicking No removes the question dialog box and leaves the online program unchanged.

All pending changes are removed from the program. If the program was in Test Mode, it is removed from this mode.

5.9 About How Changes Made to Online Programs Are Verified

Once you commit any online changes after accepting the rungs, the Editor verifies the changes before they are downloaded to the Processor. The changes are verified using the same rules that are used for offline programs. All accepted online changes must be successfully verified before they are installed into a Processor.

When online program changes are verified, the Editor ignores warnings and does not check against the Variable Configurator database.

The Editor generates a log file only if you have selected the Generate Log File verify option. For more information, see "Creating a Verify Error Log File," section 4.1.

If the program is not successfully verified, you must correct the errors, re-accept the rungs, and commit the changes or enter test mode. The Editor also notifies you when either of the following events occur:

If this event occurs:	Do the following:
a global variable has been defined in the program that is not in the online configuration	Step 1. Reject the changes containing the global variable(s). Step 2. Commit any other changes. Step 3. Go offline and add the new global variable(s) to the Variable Configurator database. Step 4. Load the configuration. Step 5. Monitor the ladder program again and add the logic with the global variables. Step 6. Accept and commit the changes.
the amount of memory for the accepted online changes exceeds the amount of memory reserved for online modifications	Increase the amount of memory allotted for online modifications. See "Specifying the Amount of Memory Reserved for Editing a Program Online" in Appendix D.
the online changes exceed the allowed number of variables, timers, counters, or labels	Edit the program to correct the limitation errors and re-accept the program. See "Specifying the Amount of Memory Reserved for Editing a Program Online" in Appendix D.

Tip

You can use the Verify command to check your online changes before you accept them.

5.10 Viewing an Online Program as It Looked Before It Was Edited

While you are editing an online program and before you commit any online changes, you can view the same program as it appeared before you began making modifications. The Editor displays the original program in a separate window. The text "(Original Program)" printed before the file name in the window's title bar helps you identify the original program from the edited version. Use the Show Original Program command to display an original version of the program you are currently editing online.

To view an online program as it looked before it was edited online

Step 1. Make sure the online-edited program for which you want to view the original version is in the active online program window.

Step 2. Do one of the following:

- From the Online menu, choose Show Original program.

  or

- Click on 

In the Original Program window, power flow and rung numbers are displayed as they appear in the online edited version. If a rung was deleted, the corresponding rung in the Original Program window is marked with a "D" revision mark and displayed using the color defined for modified rungs.

The Editor updates the original program window with any changes you make to the online program. For example, if you insert a rung in the online program, the rung numbers in the original program are updated. However, you cannot make any changes to the original program; it is a read-only file.

When you cancel all changes made to an online program, this program then matches the original version. When you commit changes made to an online program, the Editor updates the version displayed in the original program window to match the one in the online program window.

5.11 Capturing the State of Logic in an Online Program

While monitoring a program, you can freeze or capture the power flow of a given rung using the Capture Trigger command from the Online menu. When the trigger is activated (becomes true), the state of the rung is captured and displayed. The rung's power flow information is not updated on the computer; however, it continues to run on the Processor. The Editor obtains trigger information from the Processor. You can capture a rung's status based on one of the following conditions:

- a coil turns on
- a coil turns off
- an error occurs with an instruction in a rung

If more than one coil exists, the coil used for the trigger is a rung's upper-right-most coil. Once a trigger is set, it is active on the next program scan. Triggers are edge-sensitive. (For example, a coil-on trigger is activated the next time that the coil goes true.)

Only one trigger can be set for a rung, but you can set multiple triggers in an online program. The only limitation is the amount of available Processor memory.

Triggers are local to the computer that set the trigger. Other online connections cannot view these triggers until they are activated.

A trigger remains set as long as the rung is displayed in the online program window. Once the triggered rung scrolls off the screen, the trigger is cleared. Also, closing an online program window clears all the triggers set in the program.

Rungs that have triggers set are indicated by a letter displayed in the rung status area representing the trigger type, either an "o", "f", or "e". When a trigger is activated, this indicator is replaced by a "T." If a rung is marked by a revision mark, the trigger indicator replaces the revision mark until the trigger is cleared.

This table summarizes some points to keep in mind if you set triggers on rungs that already contain a trigger.

If a rung already has a trigger set and you:	**Then the Editor:**
• modify the rung, or • scroll the rung off the screen, or • close the program window	clears the trigger
select the same trigger type for a new trigger	resets the trigger
select a different trigger type for a new trigger	clears the current trigger and assigns the new trigger to the rung

The Editor notifies you when another online connection has set a trigger for the same rung.

5.11.1 Capturing the State of Logic Based on a Coil Becoming True

You can freeze the power flow of a rung based on its upper-right-most coil turning on (becoming true). You use the Coil On command from the Capture Trigger menu. When a rung has a Coil On trigger set, the rung is marked with a letter “o” in the rung status area.

To set a Coil On trigger

Step 1. Select the rung or at least one instruction in the rung for which you want to set a trigger. See “Selecting Rungs,” section 2.2, or “Selecting Instructions,” section 2.7, for more information.

Step 2. From the Online menu, select Capture Trigger. Move the mouse to the right to display the Capture Trigger menu.

Step 3. From the Capture Trigger menu, choose Coil On.

When the coil goes true, “T” is displayed in the status area and power flow stops updating. The state of the rung when the coil went true is displayed until the trigger is reset or cleared.

5.11.2 Capturing the State of Logic Based on a Coil Becoming False

You can freeze the power flow of a rung based on its upper-right-most coil turning off (becoming false). You use the Coil Off command from the Capture Trigger menu. When a rung has a Coil Off trigger set, the rung is marked with a letter “f” in the rung status area.

To set a Coil Off trigger

Step 1. Select the rung or at least one instruction in the rung for which you want to set a trigger. See “Selecting Rungs,” section 2.2, or “Selecting Instructions,” section 2.7, for more information.

Step 2. From the Online menu, select Capture Trigger. Move the mouse to the right to display the Capture Trigger menu.

Step 3. From the Capture Trigger menu, choose Coil Off.

When the coil goes false, “T” is displayed in the status area and power flow stops updating. The state of the rung when the coil went false is displayed until the trigger is reset or cleared.

5.11.3 Capturing the State of Logic Based on a Rung Error

You can freeze the power flow of a rung based on a run-time error occurring within the rung. Run-time errors may be those caused by an instruction. You use the Rung Error command from the Capture Trigger menu. When a rung has a Rung Error trigger set, the rung is marked with a letter “e” in the rung status area.

To set a Rung Error trigger

Step 1. Select the rung or at least one instruction in the rung for which you want to set a trigger. See “Selecting Rungs,” section 2.2, or “Selecting Instructions,” section 2.7, for more information.

Step 2. From the Online menu, select Capture Trigger. Move the mouse to the right to display the Capture Trigger menu.

Step 3. From the Capture Trigger menu, choose Rung Error.

When a run-time error occurs for an instruction in a rung, “T” is displayed in the status area and power flow stops updating. The state of the rung when the error occurs is displayed until the trigger is reset or cleared.

5.11.4 Waiting for Triggers While Continuing To Edit

You can wait for triggers while continuing to work with the Editor. After setting any triggers, you can minimize the program’s window or open a new window for the program.

You can also display triggers in a program using two computers. Use one computer to set and display the triggers for an online program while using another computer to monitor the program, including scrolling through the same online program.

5.11.5 Re-Activating a Trigger

You can reset an activated trigger on a selected rung. This re-initializes a rung’s frozen trigger.

To re-activate a trigger

Step 1. Select the rung or at least one instruction in the rung containing the trigger you want to reset. See “Selecting Rungs,” section 2.2, or “Selecting Instructions,” section 2.7, for more information.

Step 2. From the Online menu, select Capture Trigger. Move the mouse to the right to display the Capture Trigger menu.

Step 3. From the Capture Trigger menu, choose Reset Trigger.

5.11.6 Clearing a Trigger

You can remove a set trigger for a selected rung. Once the trigger is removed, the rung's status area returns to the state it was in prior to the trigger being set.

To clear a trigger

Step 1. Select the rung or at least one instruction in the rung containing the trigger you want to remove. See "Selecting Rungs," section 2.2, or "Selecting Instructions," section 2.7, for more information.

Step 2. From the Online menu, select Capture Trigger. Move the mouse to the right to display the Capture Trigger menu.

Step 3. From the Capture Trigger menu, choose Clear Trigger.

The Editor also clears triggers when the selected rung is scrolled out of view on the screen or when an online program is closed.

5.12 Setting and Forcing Variables in Ladder Programs

You can set and force variables in ladder programs to put them temporarily in a known state for debugging purposes. To set, force, and unforce variables, you must have an online program window open. You can set or force any global variable in the configuration or any local variable present in a task that is loaded into the rack.

IMPORTANT

You can force only simple variables. You cannot force element-indexed or bit-indexed variables. For example, you cannot force variables like: vat.13, array_var[11], array_var[index_name], array_var[11].12, or array_var[index_name].bit_name.

WARNING

THE SET AND FORCE FUNCTIONS BYPASS CONTROL OF THE APPLICATION PROCESS BY THE APPLICATION PROGRAMS. IT IS THE RESPONSIBILITY OF THE USER TO DETERMINE THE POTENTIAL HAZARDS INVOLVED. FAILURE TO OBSERVE THESE PRECAUTIONS COULD RESULT IN BODILY INJURY.

WARNING

VARIABLES AND OUTPUTS THAT ARE FORCED BEFORE AC POWER IS LOST WILL REMAIN FORCED WHEN AC POWER IS RESTORED. SHOULD AC POWER BE LOST WHILE VARIABLES ARE FORCED, THE USER MUST ENSURE THAT UNEXPECTED MACHINE MOVEMENT DOES NOT OCCUR WHEN AC POWER IS RESTORED. FAILURE TO OBSERVE THESE PRECAUTIONS COULD RESULT IN BODILY INJURY.

About Setting a Variable

To set a variable is to write a value to it that can be over-written by an instruction in the same program, another program referencing the same variable, or an external device if the variable is mapped to an input. It is possible for a variable that is set to be over-written on the next scan of the program. Forced values cannot be set.

The Editor stores and displays up to 64 set variables for the current editing session. The Editor does not store the list of set variables after the current editing session has been completed. When you set a variable, it is added to the bottom of the list. Once the 65th variable is added to the set list, the variable present at the top of the set list is removed from the list. Duplicate entries are removed from the list.

When setting variables that use variable initialization, keep these points in mind:

- Setting a variable using Retained Value initialization modifies the variable's retained value, and this value becomes its new initial value.
- Setting a variable using User Specified Value initialization changes the variable's current value but not its initial value.

About Forcing a Variable

To force a variable is to write a value to it that cannot be over-written by an instruction or another program referencing the same variable. Only unforcing the variable permits its value to be changed. Up to 64 variables can be forced. Forced variables are stored on force pages, a maximum of four, each containing 16 variables.

About the Set/Force/Unforce Dialog Box

Both setting and forcing operations for ladder programs are selected from the Set/Force/Unforce dialog box. This dialog box can be displayed in both a basic and an expanded form and is associated with only one rack connection at a time. When you choose the Set/Force/Unforce option from the Online menu, the basic Set/Force/Unforce dialog box is displayed. This basic dialog box takes up less room on the screen than the expanded dialog box so that you can view more of your program. The expanded dialog box takes up more room, but has more options.

You can switch back and forth between the basic and expanded dialog boxes using the More button in the basic dialog box and the Less button in the expanded dialog box.

To set, force, or unforce variables, you must have an online program window open. The same Set/Force/Unforce dialog box is used for each online program window using the same rack connection. Opening a program window using a different rack connection disables all fields on the Set/Force/Unforce dialog box, except the force list, that applied to the previous rack connection. Only the force list is actively updated on the Set/Force/Unforce dialog box for the online program window(s) using the previous rack connection. You can unfreeze this dialog by activating the online window that is used for the other rack connection. If you choose the Set/Force/Unforce command from the online program window with the new rack connection, the Set/Force/Unforce dialog box is updated to reflect the information for the new connection.

When you select an offline program window or the Verify output window, the Set/Force/Unforce dialog box remains in its current state.

5.12.1 Setting Variables

To set a variable is to write a value to it that can be over-written by any of the following:

- an instruction in the same program
- another program referencing the same variable
- by an external device if the variable is mapped to an input

You can set any global variable in the configuration or any local variable present in a task that is loaded into the rack.

To set a variable

Step 1. Make sure an online program window is open.

Step 2. Do one of the following:

- Click on 

  or

- From the Online menu, choose Set/Force/Unforce.

  The Set/Force/Unforce Variables dialog box is displayed. You should keep this dialog box open while you are setting, forcing, or unforcing multiple variables.

Step 3. If you want to set a local variable, enter the name of the program in which the variable is used in the Program field. If the variable is a global one, leave the field blank.

Step 4. Enter the name of the variable to be set. Only simple variables and array elements can be set.

If you had a simple variable selected before you chose the Set/Force/Unforce command, this variable is the default. If no variable was selected, the last variable set, forced, or unforced is the default.

Step 5. In the Value/State field, enter the value or state to set.

For Boolean variables, the valid range is on, off, true, false, T, F, 1, or 0. You can enter either decimal or hexadecimal values for non-Boolean variables.

Step 6. Click Set to set the variable.

Note: *A value displayed in the Value/State field is the last value set; it is not the current value. The value in this field is not monitored.*

Tip

To quickly fill in the Program and Variable Name field for a local variable you want to force, select the variable in the program before choosing the Set/Force/Unforce command. Selecting a global variable in the program before choosing the Set/Force/Unforce command fills in the Variable Name field in the Set/Force/Unforce dialog box.

Tip

Selecting a variable from the Set Variables List fills the parameters within the Current Selection group box with the information for the variable.

5.12.2 Forcing Variables

To force a variable is to write a value to it that cannot be over-written by another program referencing the same variable. You can force any simple global variable in the configuration or any simple local variable present in a task that is loaded into the processor. Only unforcing a variable permits it to be changed by a program or an external device.

IMPORTANT

You can force only simple variables. You cannot force element-indexed or bit-indexed variables. For example, you cannot force variables like: vat.13, array_var[11], array_var[index_name], array_var[11].12, or array_var[index_name].bit_name.

To force a variable

Step 1. Make sure an online program window is open.

Step 2. Do one of the following:

- Click on 

  or

- From the Online menu, choose Set/Force/Unforce.

  The Set/Force/Unforce Variables dialog box is displayed. You should keep this dialog box open while you are setting, forcing, or unforcing multiple variables.

Step 3. If you want to force a local variable, enter the name of the program in which the variable is used in the Program field. If you want to force a global variable, leave the field blank.

Step 4. Enter the name of the variable to be forced.

Only simple variables can be forced. The simple variable selected before choosing the Set/Force/Unforce command is the default. If no variable was selected, the last variable forced is the default.

Step 5. In the Value/State field, enter the value or state to force.

For Booleans, enter true, false, T, F, on, off, 1, or 0. You can enter either decimal or hexadecimal values for non-Boolean variables.

Step 6. Select the force page (1-4) on which to store the variable using the arrows in the Force Page field.

Specifying the force page lets you control how forced variables are grouped. Variables grouped on one force page can be unforced at the same time.

Step 7. Click Force to force the variable.

Forced variables change to the color you specified on the Colors tab. The default color for forced variables is red.

Tip

To quickly fill in the Program and Variable Name field for a local variable you want to force, select the variable in the program before choosing the Set/Force/Unforce command. Selecting a global variable in the program before choosing the Set/Force/Unforce command fills in the Variable Name field in the Set/Force/Unforce dialog box.

Tip

Selecting a variable from the Force Variables List fills the parameters within the Current Selection group box with the information for the variable.

Tip

You can also view the force table in the Expanded Set/Force/Unforce dialog box.

5.12.3 Unforcing Variables

Unforcing a variable allows it to be written to by any of the following:

- another instruction in the program
- another program
- an external device if the variable is mapped to an input

You can unforce any global simple variable or any local simple variable present in a task that is forced.

To unforce a variable from the Variables dialog box

- In the Set/Force/Unforce Variables dialog box, enter the variable and click Unforce. The selected variable must already be forced before you can unforce it.

To unforce a variable

Step 1. Make sure an online program window is open.

Step 2. Do one of the following:

- Click on 

  or

- From the Online menu, choose Set/Force/Unforce.

  The Set/Force/Unforce Variables dialog box is displayed. You should keep this dialog box open while you are setting, forcing, or unforcing multiple variables.

Step 3. If you want to unforce a local variable, enter the name of the program in which the variable is used in the Program field. If you want to unforce a global variable, leave the field blank.

Step 4. Enter the name of the variable to be unforced. Only simple variables can be forced. The simple variable selected before choosing the Set/Force/Unforce command is the default. If no variable was selected, the last variable set, forced, or unforced is the default.

Step 5. Click Unforce to unforce the variable.

Variables grouped on one force page can be unforced at the same time. See “Unforcing an Entire Force Page,” section 5.12.7.

Tip

Selecting a variable from the Force Variables List fills the parameters within the Current Selection group box with the information for the variable.

5.12.4 About the Expanded Set/Force/Unforce Dialog Box

Additional options not available in the basic Set/Force/Unforce Variables dialog box can be displayed by clicking More in the dialog box. Use the resulting expanded dialog for:

- Viewing the Total Number of Forced Variables
- Viewing a List of Set or Forced Variables
- Unforcing an Entire Force Page

You can toggle back and forth between the basic and expanded dialog boxes using the More and Less buttons.

5.12.5 Viewing the Total Number of Forced Variables

The Forced Variables group in the expanded Set/Force/Unforce variables dialog box allows you to view the total number of forced variables (Total Forced), which can be a maximum of 64. The total forced is useful if you want to force a large number of variables and need to keep track of how many more you can force or need to determine if forced variables are present on other force pages.

5.12.6 Viewing a List of Set or Forced Variables

You can view a list of set or forced variables in the list box of the expanded Set/Force/Unforce dialog box.

To view a list of set or forced variables

Step 1. Make sure an online program window is open.

Step 2. From the Online menu, choose Set/Force/Unforce. The Set/Force/Unforce dialog box is displayed.

Step 3. Click More.

Step 4. In the expanded Set/Force/Unforce dialog box, click Set Variables or Forced Variables. The list of variables for this edit session is displayed in the List box. Up to 64 set variables are displayed on a set variables list. Up to 64 variables are stored on a forced variables list (16 per force page). You can view only one page at a time.

5.12.7 Unforcing an Entire Force Page

You can unforce an entire page of forced variables. This option is useful when you have grouped related forced variables together on one page.

To unforce an entire force page

Step 1. Make sure an online program window is open.

Step 2. From the Online menu, choose Set/Force/Unforce. The Set/Force/Unforce dialog box is displayed.

Step 3. Click More.

Step 4. In the expanded Set/Force/Unforce dialog box, use the Current Page field to choose the force page you want to unforce.

Step 5. Click Unforce Page. The entire page is unforced.

5.12.8 Testing If Variables in the Rack Are Forced

Programs that are running can test if any variables in a rack are forced. The logic can access the reserved global variable FORCINGSTATUS@. This variable will be true if any variables in the rack are forced.

5.13 Viewing and Clearing Run-Time Errors

You can view any run-time errors that occur in an online program that you are monitoring. Use the Error Log to view error messages from the task information log. You can access this log from the Program Properties. The error log records the first, second, and last (most recent) error that occurred in the program. The last error number and its text message is displayed along with the rung number in which the error occurred. Information about any bus errors (Error 31) is also displayed.

For specific information about run-time error codes, see the AutoMax Enhanced Ladder Language Reference Manual, J2-3094.

To view run-time errors

Step 1. Make sure the online program for which you want to view run-time errors is the active program.

Step 2. Access Program Properties dialog box in one of following ways:

- With no program items selected, choose Properties from the File menu.

  or

- With no program items selected and the mouse pointer resting on the grid area away from any instructions or wires, press the right mouse button to display the pop-up menu. From the pop-up menu, choose Properties.

Step 3. Choose the Error Log tab.

To clear the error log

Step 1. Access the Error Log tab from the Program Properties dialog box. See step 2 above for more information.

Step 2. Click Clear Errors.

Step 3. Click OK.

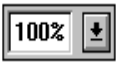
Appendix A

Toolbars, Palettes, and the Status Bar

A.1 About the Standard Toolbar

The Standard toolbar contains shortcut buttons to many menu commands, such as cut, copy, paste, save, and print. It is located by default at the top edge of the program window, but you can move it to another location. By pausing the pointer over a toolbar button, you can display a brief description of the button.

You can choose to display or hide the toolbar by choosing Toolbars/Palettes from the View menu and selecting or deselecting Standard.

As a shortcut for:	Click this button:
Opening a program	
Saving a program	
Printing a program	
Cutting an object	
Copying an object	
Pasting an object	
Finding a variable or instruction	
Verifying a program	
Listing program variables and properties (Variable List)	
Changing the display size of a program (Zoom)	
Connecting online to a Processor	
Accepting changes to an online program	

As a shortcut for:	Click this button:
Quitting test mode	
Setting/Forcing/Unforcing a variable	
Showing the original copy of an online program	
Pausing an online program	
Accessing context-sensitive help	

A.2 About the Ladder Language Toolbar

The Ladder Language toolbar provides access to the instruction palettes. Each button on the toolbar provides access to a certain group of instructions, such as relay or arrays. It is located by default under the Standard toolbar, but you can move it to another location.

You can choose to display or hide the toolbar by choosing Toolbars/Palettes from the View menu and selecting or deselecting Ladder Language.

By pausing the pointer over a button, you can display a brief description that identifies the instruction palette associated with the button.

To access these instructions:	Click on this button:
relay	
timers and counters	
shift and move instructions	
comparison	
math	
array	
program control, immediate input and output, and I/O read and write	

A.3 About the Instruction Palettes

You access an instruction palette by pointing and clicking on a Ladder Language toolbar button. Once the instruction palette is displayed, you drag instructions from it and insert them into a program. Instructions that perform similar operations are grouped together on a palette. For example, all relay instructions are grouped together on the relay palette.

If you frequently use instructions from a particular palette, you can choose to keep it open. You can then just choose an instruction and drag it into your program without having to click on the ladder toolbar palette. You can move opened palettes into the program area where it becomes a floating palette or anchor them along any screen area's edge.

The available palettes are:

Relays

- Normally Open Contact (NOI)
- Normally Closed Contact (NCI)
- Positive Transition Contact (PTI)
- Negative Transition Contact (NTI)
- Always True Contact (ATI)
- Always False Contact (AFI)
- Coil (CO)
- Set (Latch) Coil (SCO)
- Reset (Unlatch) Coil (RCO)

Timers/Counters

- Timer On Delay (TON)
- Timer Off Delay (TOF)
- Timer Pulse (TP)
- Retentive Timer On (RTO)
- Count Up Down (CTUD)

Shift/Move

- Shift Left (SL)
- Shift Right (SR)
- Circular Rotate Bits Left (ROL)
- Circular Rotate Bits Left on Transition (RL)
- Circular Rotate Bits Right on Transition (RR)
- Circular Rotate Bits Right (ROR)
- Move Bits Between Integers and Double Integers (MVB)
- Move Source Data to Destination (MOVE)
- Masked Move (MVM)

Comparison

- Equal To (EQ)
- Greater Than Or Equal To (GE)
- Greater Than (GT)
- Less Than Or Equal To (LE)
- Less Than (LT)
- Limit (LIMIT)
- Mask Compare (MSK)
- Not Equal To (NE)

Math

- Absolute Value (ABS)
- Add (ADD)
- Divide (DIV)
- Modulo (MOD)
- Multiply (MUL)
- Multiply Divide (MDV)
- Negate (NEG)
- Square Root (SQRT)
- Subtract (SUB)
- Logical AND (AND)
- Logical NOT (NOT)
- Logical OR (OR)
- Logical Exclusive XOR (XOR)
- Convert Integer Data to BCD (TO_BCD)
- Convert From BCD to Integer Data (BCD_TO)

Arrays

- Unary Array Operations (AR1)
- Multi-Array Operations (AR2)
- Array Compare (ARC)
- Array Shift Down (ASD)
- Array Shift Up (ASU)

Miscellaneous

- Set Event (SET)
- Jump (JMP)
- Label (LBL)
- I/O Read (IOR)
- I/O Write (IOW)
- Immediate Input (IN)
- Immediate Output (OUT)

A.4 Moving Toolbars and Palettes

You can move the standard and ladder language toolbars and the instruction and paste palettes from their default position to any location to suit your working style.

You can move a toolbar or palette to any edge of the screen where it becomes anchored at the new location. You can also move a toolbar or palette into the program window where it becomes a floating toolbar, which you can move and close like a window.

To move a toolbar or palette

Step 1. Place the pointer on any space between the buttons of the toolbar or around the toolbar or palette.

Step 2. Press down the left mouse button and drag the toolbar or palette away from its current location. The toolbar or palette becomes an outlined box.

Step 3. Move the toolbar or palette to any screen edge or to a location within the program window and release the mouse button.

When you:	The toolbar or palette becomes a:
anchor a toolbar or instruction palette at the left or right edge of the screen	vertical toolbar or palette
anchor a toolbar or instruction palette at the top or bottom of the screen	a horizontal toolbar or palette
move the toolbar or palette to a location inside the program window	a floating toolbar or palette

To anchor a floating toolbar or palette

Step 1. Place your pointer on the edge of the toolbar or palette and press the left mouse button.

Step 2. Drag the toolbar or palette to any edge of the screen and release the mouse button.

Tip

Close a floating toolbar or palette by clicking on  , located in the upper right-hand corner of the toolbar or palette.

Tip

If you move a toolbar or palette out of the program window or just want to restore the toolbars and palettes to the Editor's default, click on the Restore Defaults button in the Toolbar and Palette dialog box.

A.5 About the Status Bar

The status bar is located at the bottom of the Editor window and displays information about the program objects you have selected and the state of the program in the active window. Look at the status bar for helpful messages as you work with your programs.

Verified

For the active program window, the status bar displays VERIFIED when the program has been successfully verified and is ready for loading in to the Processor.

Not Verified

If the program has not been successfully verified or changes have been made to it since the last verify operation, the status bar displays NOT VERIFIED.

Run or Stop

For the active, online programs, the status bar shows if the program is running (RUN) or stopped (STOP).

Help for the Selected Menu

The status bar shows abbreviated help for the selected menu item in the active window.

Name and Description of Selected Object

For the active window, the status bar can display any of the following:

If this is selected:	Status bar displays the:
single contact or coil	associated parameter and the primary variable description
single block	parameter and description for the primary output
single rung	first line of the associated rung description

Appendix B

Keyboard Shortcuts

Here is a listing of keyboard shortcuts for the Editor.

To:	Do the following:
Open a program	Press CTRL+O
Save a program	Press CTRL+S
Print a program	Press CTRL+P
Access properties for a program item	Press F2 Rungs, instructions, and variables must be selected first
Move to the next field	Press TAB
Move to the previous field	Press SHIFT+TAB
Select all the rungs in a program	Press CTRL+A
Cut rungs, instructions, or text	Select the item(s) to be cut and press CTRL+X
Copy rungs, instructions, or text	Select the item(s) to be copied and press CTRL+C
Paste text	Press CTRL+V
Paste rungs or instructions	Press CTRL+V Drag the logic from the Paste Palette
Clear rungs, instructions, or text	Select the item(s) to be cleared and press DEL
Go to a rung	Press CTRL + G
Turn on Rung Descriptions Only	Press F5
Access the Zoom dialog box	Press F7
Access the Find dialog box	Press CTRL+F
Access the Replace dialog box	Press CTRL+R
Find the next match	Press SHIFT+F4
Find the previous match	Press SHIFT+F5
Access the Online Task Manager	Press F9
Monitor a PC program	Press F10
Quit Test Mode	Press CTRL+Q
Access the application window control menu	Press ALT+Spacebar
Access the program control menu	Press ALT+ – (dash)
Close the active window	Press CTRL+F4
Switch to the next program window	Press CTRL+F6
Close the Editor	Press ALT+F4

Appendix C

Using the Pre-Defined (Reserved) Ladder Language Variables

The Editor contains pre-defined variables that you can use in an individual ladder program to:

- execute logic based on a Processor scan
- specify how to handle error conditions
- check scan time execution

The pre-defined ladder language variables are local variables, which you can use for ladder instruction parameters. Their names are reserved and appear in the choices offered by the Variable Smart Matching option.

Using the Pre-Defined Program Scan Variables

Use the following Boolean variables to execute logic based on the Processor's scan. Only use these variables for input parameters (read-only):

• first_scan	Use this variable to execute logic during a program's first scan. This variable is true during the initial scan of the ladder rung and false during all other scans.
• second_scan	Use this variable to execute logic during a program's second scan. This variable is true during the second scan of the ladder rung and false during all others.
• last_scan	Use this variable to execute logic during a program's last scan. This variable is true on the final pass of the ladder rung after you have selected TASK STOP. This variable is not set when a STOP ALL error occurs. To use the last_scan variable, your program scan must be less than 0.5 seconds. **IMPORTANT** Do not use the last_scan variable in an event-driven program. Because these programs are based on the occurrence of an event, the last_scan variable may never be executed when used in an event-driven program.

Using the Pre-Defined Error Handling Variables

Use the following Boolean variables to help you handle error conditions. Use error_eno and no_error_log for output (read and write) parameters. You can use task_error as either an input (read-only) or output (read and write) parameter:

• task_error	This variable is set true whenever an error is found. Monitor the bit to see if an error occurs during execution and clear it by using the ladder logic. This bit is set true even if errors are not being logged.
• error_eno	Use this bit to determine the value ENO outputs will have if the instruction has an error. The default value is false, which disables any instructions that are connected to the ENO output, possibly making math expressions incomplete. When you set error_eno true, you can continue the execution of the logic connected to the ENO output even if the instruction block had an error. This variable can be changed during ladder logic program execution.
• no_error_log	Set this variable true to prevent errors from being entered into the program error log or being seen by the rung monitor. Only one error is logged for each instruction per program scan; however, you may want to prevent the errors encountered on certain instructions from being entered into the error log. The default state for the no_error_log is false. You can suppress error messages for a group of rungs by changing this variable during program execution. STOP ALL errors and parameter limit errors for AR1, AR2, and ARC instructions are still inserted into the error log when the no_error_log variable is true.

Using the Pre-Defined Ladder Execution Time Variables

Use the following double integer variables to help you check and monitor the program's execution time. Only use these variables as input parameters.

• task_usec_max	Use this variable to monitor the maximum execution time (in μs) for the current program.
• task_usec_now	Use this variable to monitor the latest execution time (in μs) of the current program. The execution time is the real "clock" time it took the program to run from start to finish. It includes the execution time for higher priority programs and interrupt service routines if they run while your program is running.

To reset these times, write a value of 0 into the variable.

Appendix D

Online Editing Memory Limits

Limits exist for the amount of memory reserved for editing programs online. The Editor reserves a default 4096 bytes for editing programs online. However, you can change the amount of memory reserved for online editing. See “Specifying the Amount of Memory Reserved for Editing a Program Online” in this Appendix for more information.

Each rung that was modified or added during an online editing session is checked against the allocated amount of reserved memory when you commit the changes or enter test mode (that is, when you download the rungs to the Processor). If the total number of online changes exceeds the allocated memory, a message box appears to notify you. To continue editing a program online once these limits have been exceeded, save the program to the PC and re-download it to the Processor.

Limitations for Editing Programs Online

You cannot add more than:

- 32 variables
- 8 timers and/or counters
- 16 labels

These limits are checked after you choose Accept from the Online menu. If the changes exceed any limit, the Editor notifies you of the location at which the problem(s) occurred. The changes are not downloaded to the Processor. You can then edit the program to correct the limitation errors and re-accept the program.

Restrictions for Editing Programs Online

While editing a program online you cannot add these types of variables:

- a global variable that is not already used in the configuration
- new or existing array variables
- an element of an array that is larger than the defined Maximum Array Index
- new Software Events

Other restrictions for editing programs online:

- You can change all the properties for a newly added variable, but you cannot add or change a variable description.
- You cannot inline edit variable or rung descriptions.
- You cannot change any Program Properties.

Some instructions use internal buffers to perform their operation. Consequently, the state of the following instructions’ operation is stored from program scan to program scan:

- Counter instruction (CTUD)
- Array instructions (AR1, AR2, ARC, ASU, ASD)
- Shift Register instructions (SL, SR, RL, and RR)
- Set Event instruction (SET)

During online editing, the state information is only transferred if you have not added, moved, modified, or deleted any of the instructions. The Editor warns you when state information is not transferred.

Specifying the Amount of Memory Reserved for Editing a Program Online

You can increase or decrease the amount of memory reserved for changes made to an online program by using the Program Info tab of the Program Properties. You must change the program's properties in an offline editing session of the program. The maximum amount of memory you can reserve is 20480 bytes; the minimum is 2048 bytes.

To specify the amount of memory reserved for editing a program online

Step 1. Make sure you are editing the program **offline**.

Step 2. Access the Program Properties. See "About Program Properties."

Step 3. From the Program Info tab, enter the amount of bytes you want to reserve for online changes in the Online Reserved Memory field. Enter a value between 2048 bytes and 20480 bytes. The default amount is 4096 bytes.

Step 4. Click OK to accept the change.

Appendix E

Rung Execution Order

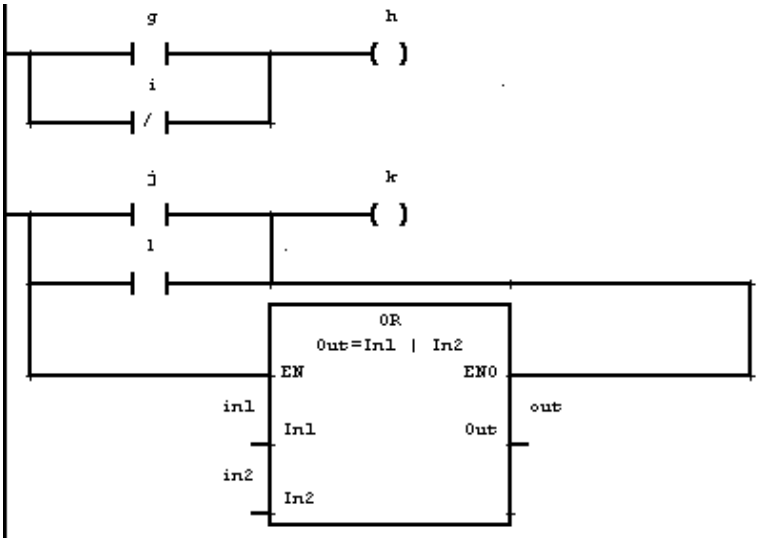
In a ladder logic program, the rungs are executed from the top to the bottom. Within a rung, instructions are executed left to right until a branch is encountered. Therefore, in a program without branches, the instructions are executed left to right in a rung, with the first rung being executed first and the last rung executed last.

Keep in mind that using JMP instructions within a ladder program changes the rung execution sequence. For more information about using the JMP and LBL instructions, see the AutoMax Enhanced Ladder Language Reference manual, J2-3094.

Using branching techniques in a rung changes the sequence in which the instructions are executed. Parallel branches are executed from the bottom up. This means that the bottom branch is executed first and the top branch last. How branches are executed are listed below:

Branching technique #1:

Connecting contacts or block instructions acting as inputs in parallel (OR condition)



How the branches are executed:

How parallel branches of input instructions execute depend upon whether:

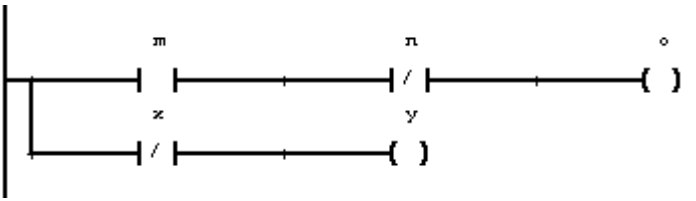
- The relay input instructions use simple variables, bit−indexed integer, double integer variables, or element−indexed array variables
- Block instructions are used in a parallel branch

This table summarizes how parallel branches of inputs are executed:

If a parallel branch contains:	The instructions are executed as follows:
relay input instructions using simple variables	The logic is executed starting at the last parallel branch and going up the branches until a true row is encountered. Once a true row is encountered, any branches remaining above the true branch will not be executed.
relay input instructions, bit-indexed integer or double integer variables, or element-indexed array variables	All parallel branches are executed, starting with the last branch and working up the parallel branches.

Branching technique #2:

Connecting a logical sequence of instructions composed of inputs and an output in parallel to a rung



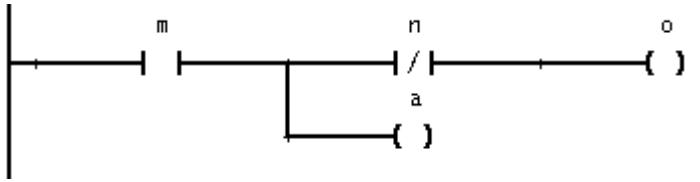
How the branches are executed:

Parallel branches made up of a sequence of instructions composed of inputs and an output are executed as follows:

- Each branch is fully executed before the main rung
- Multiple parallel branches are executed from the bottom up

Branching technique #3:

Connecting one or more coils parallel to a rung



How the branches are executed:

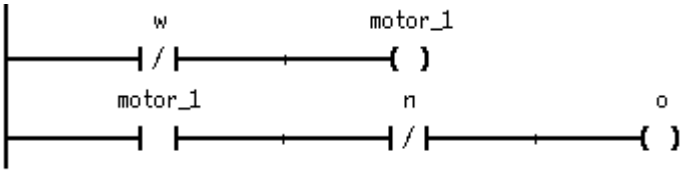
Parallel branches comprised of coils are executed as follows:

- Logic preceding the branched coil is executed, followed by the branched coils. The bottom coil of the branch is executed first and the top coil of the branch is executed last.
- Once the branched coils have been executed, the remaining logic of the main rung is executed.

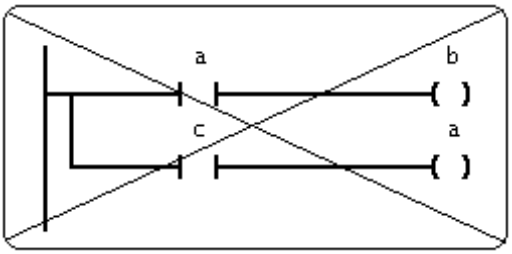
Programming Techniques

The relay output instructions (coils) and the outputs of block instructions are set when the instruction is executed. Therefore, if a variable is used as an input and as an output within the same **program**, you can use a more recent value for the input if the output instruction is executed first. For example, the variable *motor_1*

used on the NOI instruction will contain a more recent value, since *motor_1* of the coil is executed first. The *motor_1* output is set upon execution.

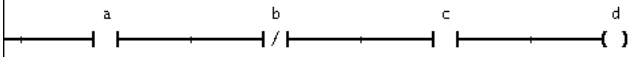


However, do not create a **rung** where the same variable is used on an input and an output, and the output is executed first. Creating such a rung results in a verify error. The following figure shows such a rung:

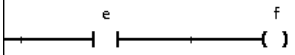


The following figure illustrates how rungs and parallel branches are executed.

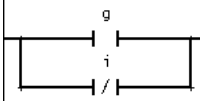
1) Rung 1 is the first rung in the program to be executed.



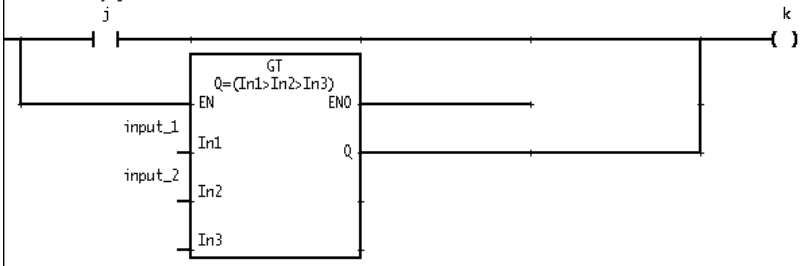
2) Rung 2 is the second rung to be executed.



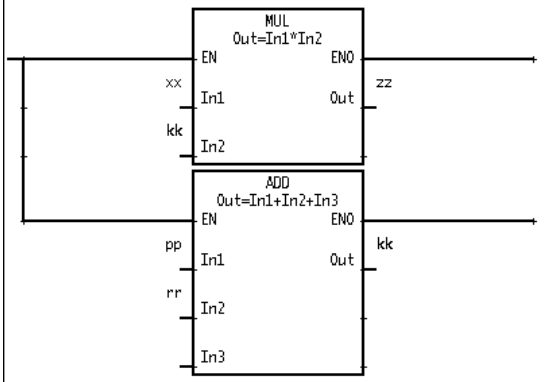
3) Rung 3 is executed next. Instuction i is executed first followed by g, if i is not true. When i is true, g is skipped and coil h is then executed.



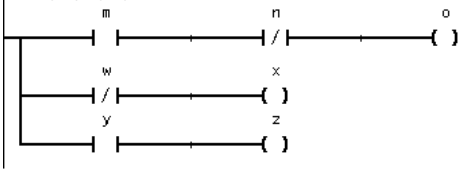
4) Rung 4 is executed fourth. The parallel branch comprised of instructions j, l and the GT instruction are all executed; the GT instruction is executed first, followed by j and then l.



5) Rung 5 is executed next. This rung is an example of how you can use a variable as an output in a parallel branch and then as an input to the branch above. The ADD block is executed first followed by the MUL instruction. Since output kk is updated after the ADD instruction executes, the newly updated value is used in the MUL instruction.



6) Rung 6 is executed last. The instructions are executed in this order: y, z, w, x, m, n, o.



Appendix F

Converting Ladder Programs Created Using an Older Version of AutoMax

You must convert any ladder program that was created using an AutoMax Executive earlier than V4.X before you can edit it using the V4.X ladder editor and the enhanced ladder instruction set. Once you convert the program, you cannot use it with an earlier PC editor.

You can convert ladder programs by doing any of the following:

- selecting a pre-V4.0 system from the System Configurator
- copying a system, section rack or program
- opening a program from the Editor

See “Opening Programs,” section 1.2.

The software prompts you to convert the system, which includes converting old ladder programs. Any conversion warnings are posted in a log file. If the programs are converted when a system, section, or rack is converted, the warnings are placed in the file UPDATE.LOG. If you convert an individual program from the Editor, the warnings are placed in the file *program_name*.LOG. You can find UPDATE.LOG in the system directory, and *program_name*.LOG in the rack directory.

At the dialog box prompting you to convert the program, choose Yes. The Editor does the following:

- replaces the old ladder instructions with the corresponding ones from the enhanced ladder instruction set
- converts any remark sequences into a rung consisting of one coil

  The coil names are those of the keywords from the old remarks. These new variables are local variables. The remark text becomes the rung description for the new rung. Any tabs in the remark sequences are converted to single space characters.
- copies any variable description text to the new program and truncates the text to 40 characters
- renumbers the rungs starting at number one
- preserves any preset values for timers and counters
- creates new timer and counter data structures by using the variable name of the current timer or counter value as the name of the new timer or counter data structure

  The EN input of the new counter instruction is connected to the power rail.
- converts the task time (in seconds) to the equivalent number of ticks as defined for the Processor within which the program is slated to run
- deletes any descriptions associated with individually defined array elements

  These descriptions are not carried over to the converted program.
- converts shift instructions to a Shift Left Instruction. See “About How Shift Instructions Are Converted into the Ladder Editor” in this Appendix.

To save the converted file, choose the Save command. If you quit the Editor or close the converted program before saving the program, the original, unconverted program is unchanged.

The Editor creates a log file if any conversion error or warning messages are generated. See “Correcting Ladder Program Conversion Warnings and Errors” in this Appendix.

Once the Editor converts the program, it renames the original program file to *file*.@PC and renames the original remark *file*.REM to *file*.@RE. You can delete these files later if you wish.

F.1 About How Shift Instructions Are Converted into the Ladder Editor

Any shift register instructions present in the old ladder program are converted to a Shift Right (SR) instruction. The logic that drives the SHIFT and DATA parameters is converted to correspond with the SR instruction’s EN and BIT parameters, respectively. The following table describes how other conditions are handled.

If the:	Then:
old instruction had a shift register length that is less than 16	The shift register length for the corresponding SR instruction is 16 (an integer). Note that all the bits within the integer are shifted. A message is posted in the file UPDATE.LOG or *program_name*.LOG.
RESET input parameter was programmed in the old instruction	A MOVE block is added to the rung. This block is driven by the logic that was connected to the RESET parameter. The MOVE instruction resets the shift register integer when the reset coil is true.
DATA and SHIFT parameters were not programmed in the old instruction	The rung is converted into a MOVE instruction and a warning is posted in the file UPDATE.LOG or *program_name*.LOG.

F.2 Correcting Ladder Program Conversion Warnings and Errors

This section lists some of the warning messages generated when you convert an old ladder program to a V4.0 or later program and describes what you can do to fix them. The messages are posted in the file UPDATE.LOG or *program_name*.LOG.

Message:	**What the message means:**	**Resolution:**
In old Rung number *xx* (new Rung number *xx*), the old global preset variable is no longer used.	The variable assigned to the PN parameter in the old timer or counter instruction was a global integer variable.	To use a global variable as a timer or counter preset, define the timer or counter data structure as a global, double integer array by using the Variable Configurator. For more information about preset values, see “About Timer Variables” and “About Counter Variables.”
In old Rung number *xx* (new Rung number *xx*), the old global current value variable is used.	The variable assigned to the CN parameter in the old timer or counter instruction was a global integer variable.	To use a global variable to store a timer’s or counter’s current value, define the timer or counter data structure as a global, double integer array by using the Variable Configurator. For more information about current values, see “About Timer Variables” and “About Counter Variables.”
In old Rung number *xx* (new Rung number *xx*), a preset variable is used that was also used on another rung.	In older ladder programs, you could use the same preset variable on multiple timers or counters in the same program. Now, each timer or counter instruction in a program must have a unique data structure assigned to it.	To use the same preset variable on multiple timers or counters, add MOVE instructions to copy the preset value from one of the timer/counter data structures to another timer/counter data structure.

Appendix G

Glossary

Accept: To approve the edit made to an online program. Rungs that have been added, deleted, or modified must be accepted and verified before they can be downloaded to a Processor.

Bit-indexed variable: A variable referencing a bit within an integer or double integer variable. For example, pump.15 references bit 15 within the integer variable ”pump.”

Commit: To allow the Editor to verify and download changes made to an online program. You must accept online changes before you can commit them. You can commit online changes immediately after you accept them or while the program is in Test Mode.

Data Structure: Data structures contain a collection of Boolean and double integer data and are used for the timer and counter data types.

Element-indexed variable: A variable referencing an element within an array variable. For example, panel[11] references an element 11 within the array variable ”panel.” An element can be a Boolean, integer, or double integer.

Global Variables: Global variables can be referenced by ladder, Control Block, or BASIC programs in a rack. Global variables can refer to memory locations, physical I/O locations, or network locations. Global memory variables can be of any data type supported by the Editor. If you type in the first letter of a variable using upper case, the default scope will be global. The names of global variables appear in upper case.

Local Variables: Local variables are those that can only be used in the program in which they are defined. No other programs can reference them. If you type in the first letter of a variable name using lower case, the default scope will be local. The names of local variables appear in lower case.

Match: As applied to the Resolve Variable Descriptions command, a global variable in a ladder program that uses the same name as one in Variable Configurator.

For example, the Editor would determine that a global variable called PUMP_STATUS used in a ladder program is a match to a global variable called PUMP_STATUS present in the Variable Configurator.

The data type of a variable is not a factor when determining whether the global variables match.

Path: The directory structure used by the AutoMax Executive is:

drive:\library\system\rack

where:

drive	is the personal computer hard drive where the Executive is stored
library	is the base directory under which all the AutoMax systems are stored
system	is the subdirectory where the system database files are stored
rack	is the subdirectory where all the rack database files and all programs for the rack are stored

The default drive and library name are specified as part of the Setup procedure for the AutoMax Programming Executive software. If you want to create a new library or change the default (selected) library or drive, you must use the Setup procedure.

Pause: Places the Editor in the Paused state so that you can use the Online Task Manager. The programs continue to run in the Processor, but their display in a program window is not updated.

Program: Task. In the Editor, the terms “program” and “task” are synonymous.

Rung status area: The gray-shaded area left of the power rail. When rung numbers, revision marks, or set triggers are displayed, they are located here.

Test Mode: Lets you actively execute rungs, but the changes made to the online program are not permanently installed in the processor.

Trigger: A trigger is a way to capture or freeze a rung’s status while monitoring the program. Once a trigger is set, the rung’s status stays frozen on the programming terminal, but the rung continues to run on the CPU.

Index

A

AutoMax window, 1-1

C

Coil Off trigger, 5-14

Coil On trigger, 5-14

Commit command, 5-7

Constants, assigning to instruction parameters, 2-10

D

Data display, 5-6

E

events, 1-8—1-9

G

Global variables, 2-14

Glossary, G-1

I

Instruction palettes
- anchoring, A-5
- moving, A-4

L

Local variables, 2-14

M

Monitor PC Program command, 5-4

P

Page breaks, 3-7

Page setup, defining, 3-5

Power flow, 5-5
- freezing or capturing, 5-13

Print options, setting, 3-4

Printing
- defining page setup, 3-5
- how Editor prints rungs, 3-6
- multiple copies, 3-8
- page breaks, 3-7
- components of printout, 3-2
- range of rungs, 3-7
- right-justifying coils, 3-8
- setting print options, 3-4

Programs
- closing, 1-3
- converting older, F-1
- data display in online, 5-6
- displaying in multiple windows, 1-4
- displaying in split window, 1-5
- editing, 2-1
  - exiting the Editor, 1-8
- editing multiple, 1-5
  - from the Editor, 1-6
  - from the Task Manager, 1-5
- editing online, 5-1
  - accepting changes, 5-6
  - cancelling changes, 5-10
  - comparing with unedited program, 5-12
  - verifying changes, 5-11
  - changes you can make, 5-1
  - downloading changes, 5-7
  - memory limits, D-1
  - open using Monitor PC Program, 5-4
  - pausing the Editor, 5-5
  - removing program, 5-10
  - saving in test mode, 5-9
  - testing programs, 5-9
- execution, 1-7
- opening, 1-2
- power flow in online, 5-5
- printing — see Printing, 3-1
- properties, 1-7
- running and stopping ladder, 1-8
- saving, 1-3
- saving automatically, 1-4
- setting scan time, 1-7
- states of online, 5-2

verifying, 4-1
creating a verify error log file, 4-2
resolving errors, 4-3
variable configurator database, 4-2
warning messages, 4-3

Revision marks, displaying and hiding, 1-6

R

Run−time errors, viewing and clearing, 5-22

Rung Error trigger, 5-15

Rung table, 3-3

Rungs
defining grid limits for, 2-2
entering descriptions of, 2-4
in the program window, 2-4
in the Rung Properties dialog box, 2-4
inserting instructions into, 2-5
instructions
assigning variables or constants to, 2-10
changing type of, 2-9
using Find and Replace, 2-9
using Instruction Properties, 2-9
connecting
by drawing wires, 2-6
using ENO output bit, 2-6
Properties dialog box, 2-8
selecting, 2-7
order of execution, E-1
printing of, 3-6
selecting, 2-2
starting, 2-1
turning the grid on and off, 2-2

S

Scan time, 1-8

Shortcuts, B-1

State of logic, capturing in an online program, 5-13
based on coil becoming false, 5-14
based on coil becoming true, 5-14
based on rung error, 5-15

Status Bar, A-5

T

Test mode, 5-9
removing online program from, 5-10
saving online program in, 5-9

Tick rate, 1-8

Toolbar, A-3, F-2
anchoring, A-5
ladder language, A-2
moving, A-4
standard, A-1

Triggers
clearing, 5-16
re-activating, 5-15
setting, 5-13
waiting for while editing, 5-15

V

Variable name table, 3-3

Variables
assigning to instruction parameters, 2-10
changing data type of, 2-13
changing scope of, 2-14
entering descriptions of, 2-11
forcing, 5-17, 5-19
unforcing, 5-20
list of set or forced, 5-21
pre−defined, C-1
error handling, C-2
ladder execution time, C-2
program scan, C-1
selecting, 2-13
set/force/unforce dialog box, 5-17, 5-21
setting, 5-17, 5-18
testing for forced, 5-22
total number of forced, 5-21
unforcing an entire page, 5-21
using Smart−Matching to enter, 2-12

Verify Error Log File, 4-2

Verify Errors, resolving, 4-3

RELIANCE CONTROLS
DOCUMENTATION IMPROVEMENT FORM

Document Number: _____

Page Number(s): _____

Comments: *(Please give chapters, page numbers or specific paragraphs that the change will affect. Include markeups from the document or attach additional pages if necessary.)*

What will this improvement suggestion provide? _____

Originator: _____ City: _____ State: _____ ZIP: _____

Company: _____ Phone: (____) _____

Address: _____ Date: _____

Technical Writing Internal Use:	**Follow-Up Action:**
Writer: _____	Date: _____

Thank you for your comments . . .

RE 1857LC Printed in U.S.A.

RELIANCE ELECTRIC

For additional information

1 Allen-Bradley Drive
Mayfield Heights, Ohio 44124 USA
Tel: (800) 241-2886 or (440) 646-3599
http://www.reliance.com/automax

www.rockwellautomation.com

Corporate Headquarters
Rockwell Automation, 777 East Wisconsin Avenue, Suite 1400, Milwaukee, WI, 53202-5302 USA, Tel: (1) 414.212.5200, Fax: (1) 414.212.5201

Headquarters for Allen-Bradley Products, Rockwell Software Products and Global Manufacturing Solutions
Americas: Rockwell Automation, 1201 South Second Street, Milwaukee, WI 53204-2496 USA, Tel: (1) 414.382.2000, Fax: (1) 414.382.4444
Europe/Middle East/Africa: Rockwell Automation SA/NV, Vorstlaan/Boulevard du Souverain 36, 1170 Brussels, Belgium, Tel: (32) 2 663 0600, Fax: (32) 2 663 0640
Asia Pacific: Rockwell Automation, 27/F Citicorp Centre, 18 Whitfield Road, Causeway Bay, Hong Kong, Tel: (852) 2887 4788, Fax: (852) 2508 1846

Headquarters for Dodge and Reliance Electric Products
Americas: Rockwell Automation, 6040 Ponders Court, Greenville, SC 29615-4617 USA, Tel: (1) 864.297.4800, Fax: (1) 864.281.2433
Europe/Middle East/Africa: Rockwell Automation, Brühlstraße 22, D-74834 Elztal-Dallau, Germany, Tel: (49) 6261 9410, Fax: (49) 6261 17741
Asia Pacific: Rockwell Automation, 55 Newton Road, #11-01/02 Revenue House, Singapore 307987, Tel: (65) 6356-9077, Fax: (65) 6356-9011

Publication J2-3093-3 - April 1998